UP AND RUNNING
SOFTWARE DEVELOPMENT

HOW WE WORK:

We are commonly asked about how our ticket system and workflows function, and this document addresses that in some detail. We hope the videos and text are helpful. If you'd prefer a real-time demonstration, we'd be happy to do that.

It's important to mention that we always adapt to how our customers like to work using the tools they'd like to use. Common systems include activeCollab, JIRA, Redmine, Pivotal Tracker, Githut, GitLab, Mantis, Bugzilla, Basecamp, FogBugz, ZenDeskt, RT, Aha, Smartsheet, EM7, and custom systems. (Some of these are not formal ticket systems used in a software development setting, but they are listed here because some of our clients have used them for that purpose.) In terms of workflows, most of our clients use Agile or Scrum or some form of those approaches; however, we're able to work how you want to work.

We have processes in place to ensure success, but within those procedures we have complete flexibility. Stakeholders, client policies, scope, resources, budgets, timelines, etc. all differ; the process for each project must adapt accordingly. Consistent customer happiness is our metric of success, the constant, not that we followed a trademarked or industry-standard process or used a particular tool to facilitate work.

OUR SYSTEM:

As a development company, we wanted complete flexibility with our system. To enable that, we implemented a customized version of activeCollab (https://www.activecollab.com/), often referred to as "aC" at least internally to Up and Running. The customizations we implemented are largely related to reporting, enforcing work standards we think are important, and enabling our administrative team to perform their accounting processes. Simply, this is the hub of our business, one that we can extend however we want given our profession. aC allows project managers, developers, and customers to interact and collaborate in real time on every aspect of a project

OUR METHODOLOGY:

A Flexible Approach: We use an Agile process by default within Up and Running, though as mentioned we can adapt to the workflows you prefer. If there is no preference, we'll usually take the following approach, customized to the specific context of the project (the work being done, how success is defined for that project, and the people involved). One of our owners is a certified ScrumMaster, and he's worked with VPs of Technology, Directors, and like senior-level technical roles at companies such as Yahoo!, GE, ScienceLogic, RichRevelance, Passare, and many more well-respected, mature software development companies or departments and startups. We've learned a lot from these great clients, and we've also been somewhat assured of our current approach we use because it's the same approach that is mandated to us by these clients.

Defining Agile: Agile means, as presented by the "Agile Manifesto" here http://agilemanifesto.org/, a results-focused approach that centers on individuals and interactions to produce working software via customer collaboration with built-in methods of handling the inevitable change and issues that arise in any long-term software development project. Here are the guiding principles of Agile: http://agilemanifesto.org/principles.html. More information: http://en.wikipedia.org/wiki/Agile_software_development

Agile, in Practice: The following will communicate what Agile is via the steps we commonly use for its implementation. This section presents the information three ways: in a nutshell, in detail, and in greater detail. You do not need to read this. It's very easy to understand once it's implemented and it's been experienced a couple of times. However, we thought we'd put it here to explain what we do at various levels of detail. Also, any of these can be elaborated on in real-time and/or via a demonstration if you'd like.

An important note is that not all projects use all of these steps. We always do what clients are comfortable with, what clients want, and, by default, what is the most efficient approach to get work done. All of this is a means to an end, which is a system that achieves your business goals.

In a nutshell:

1. Define
2. Implement a chunk of work within a set timeframe (often every 1-3 weeks)
3. Communicate, adapt, learn throughout
4. Repeat

In detail:

1. Initial Product Backlog Creation.
2. Iterations
    a. Sprint Planning Meeting Part 1 for product backlog grooming with the customer
    b. Sprint Planning Meeting Part 1 with the team
        i. Determine available Story Points
        ii. Determine estimates on backlog items without estimates using Story Points
        iii. Determine the sprint backlog
        iv. Break down stories into tasks
    c. Sprints
        i. Hold a daily standup with the team
        ii. Work on tasks
        iii. Perform QA and tests, basically creating release-ready code
        iv. Release Demo
    d. Generate the release
        i. Demo release to any interested parties
        ii. Gather feedback/additional backlog items and record them in the product backlog
    e. Retrospective with the development team
        i. Review what went well on the project and plan for use there on future sprints
        ii. Review what didn't go well and plan for ways of addressing those deficiencies
        iii. Return unfinished items back to backlog for the next iteration

In greater detail:

1. Product Backlog Grooming
    a. Purpose: Organizes and provides context around upcoming work items (the product backlog) that will be handled by the development team
    b. Duration: 60 minutes or more, prior to sprint planning sessions
    c. Objectives:
        i. Add items to the product backlog, either items on the road map or new items discovered during development

   d. Flesh out any epics (high-level objective) or stories (a form of backlog item that represents more of a specific use case or function in the software) in the system that require more breakdown
   e. Participants: Client, Project Lead, and Developers
   f. Deliverables: An updated product backlog that contains new items and additional context on existing items

2. Sprint Planning Session, Part 1
   a. Purpose: Review the items in the product backlog together, and prioritize the list
   b. Duration: 30 minutes
   c. Objectives:
      i. Prioritize the list of product backlog items so the development team knows explicitly the order in which to do the tasks
      ii. Talk and work through any last-minute details about the top-ranked items in the product backlog
   d. Participants: Project Lead and Client
   e. Deliverables:
      i. A prioritized list of items in the product backlog
      ii. Documentation of additional context for the items in the product backlog that were discussed
3. Sprint Planning Session, Part 2
   a. Purpose: Estimate out any new items in the product backlog and re-estimate those that had new context added to them. Determine the sprint backlog
   b. Duration: 60 minutes
   c. Objectives:
      i. Play estimate poker for any items in the product backlog that are new or have new context. Use Story Points for this. Here’s an online tool that can be used: http://www.planningpoker.com/
      ii. Based on the team’s velocity/allocation, remove sprint backlog items that cannot be finished in time, creating a sprint backlog
   d. Participants: Project Lead and Developers
   e. Deliverable: Sprint backlog for the sprint
4. Sprint Execution
   a. Purpose: Execute on the work (develop)
   b. Duration: 1 week or whatever the sprint time period is defined as
   c. Objectives:
      i. Execute on the items in the sprint backlog
      ii. Update the product backlog with new items and findings
      iii. Review/QA the finished code, and get client feedback if required
      iv. Meet with the team to discuss progress each day via standups to communicate what was accomplished, what will be done, and what obstacles exist if any
   d. Participants: Project Lead, Client, and Developers
   e. Deliverables:
      i. Completed working code that could be deployed if needed
      ii. A burndown that is regularly updated, of the sprint
      iii. Any remaining items that need to go back to the product backlog
5. Retrospective Meeting:
   a. Purpose: Wrap up the sprint and prepare for the next sprint

b. Duration: 30 minutes
c. Objectives:
   i. Review completed work and gather feedback. Update the product backlog if required
   ii. Review how the sprint went with the development team. Make adjustments to how the work is handled
   iii. Determine the team's new velocity
   iv. Set up items for the next sprint
d. Participants: Project Lead, Client, and Developers
e. Deliverables:
   i. A package that can be delivered. This could be code that is deployed into production or just to a staging site. Feature branches should be merged and pushed into a staging or production branch
   ii. Any changes to the process/policy of the team
   iii. The team's new velocity
   iv. A new milestone for the new sprint
   v. A new burndown chart file
   vi. Recording of the demo/review, so it can be archived and communicated

Vocabulary:

1. Burndown Chart: Represents the amount of work remaining over time. It's a downward trending line chart that the project lead produces. More information: http://www.scrumalliance.org/community/articles/2007/march/glossary-of-scrum-terms#1127
2. Epic: A form of backlog item that represents a really high-level objective. Example: a user system, supporting groups and ACLs.
3. Estimate Poker: A process where each team member determines independently how long a task would take and then everyone simultaneously shows the results. An estimate is determined from the group's consensus. More information: http://en.wikipedia.org/wiki/Planning_poker An example tool that can be used online: http://www.planningpoker.com/
4. Product Backlog: This is the main list of requirements for the project. It's comprised of items that represent units of work. We use aC's unknown milestone to hold all the product backlog items. More information: http://www.scrumalliance.org/community/articles/2007/march/glossary-of-scrum-terms#1125
5. Retrospective: A meeting where the team reviews how the sprint went and makes any changes to the process or policy on how the team works. More information: http://www.scrumalliance.org/community/articles/2007/march/glossary-of-scrum-terms#1113
6. Sprint Backlog: Represents the set of tasks that have been outlined to achieve the sprint's goal. The items in the sprint backlog are organized under a specific milestone in aC. More information: http://www.scrumalliance.org/community/articles/2007/march/glossary-of-scrum-terms#1117
7. Story: A form of backlog item that represents more of a specific use case or function in the software. Example: The user will be able to log into the website, using a username and password.
8. Story Point: A unit of measure that is unrelated to time and represents a level of work that the team can complete in the given sprint.

9. Task: A form of backlog item that represents a smaller functional unit of a story typically. Example: Write up a method in the user model that handles authentication by comparing password hashes.
10. Velocity: Represents how much backlog effort the team can get done in a sprint. This is determined, based off estimates, but over time the velocity gets averaged out and becomes an effective predictive instrument. More information: http://www.scrumalliance.org/community/articles/2007/march/glossary-of-scrum-terms#1110

WORKFLOWS:

Video Walkthrough

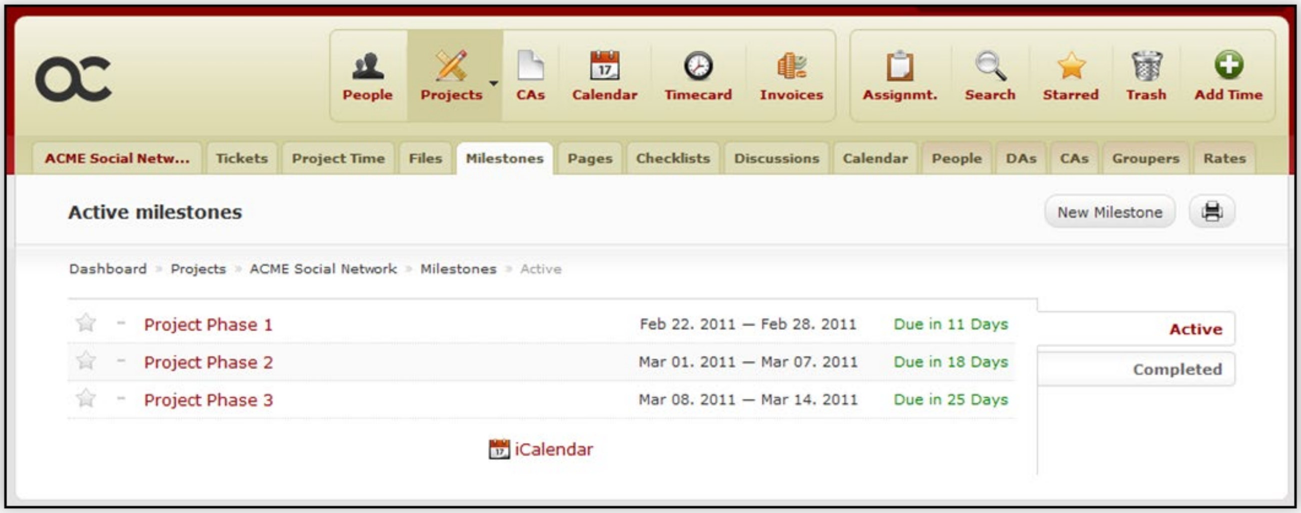
- Here is a collection of videos that present our common development workflows. This focuses on tickets, which are generally discrete tasks from the product backlog, for the sprint execution component of the Agile process. Simply, this is the definition of the work to do, and it's where a lot of communication and interaction take place for that reason.

  Site: http://www.upandrunningsoftware.com/acdemo
  Password: uar123

Ticket Screenshots

Here are some screenshots of how tickets look to give you an idea of what's in aC, as well as context regarding each screenshot. Any or none of these can be used within your project to the degree that you'd like. How work is done depends on how you like to work and what's the most efficient way to produce the results you want consistently.

Milestones: First, the project manager (who is also a senior developer and sometimes an architect within Up and Running) works with the client to create iteration milestones.

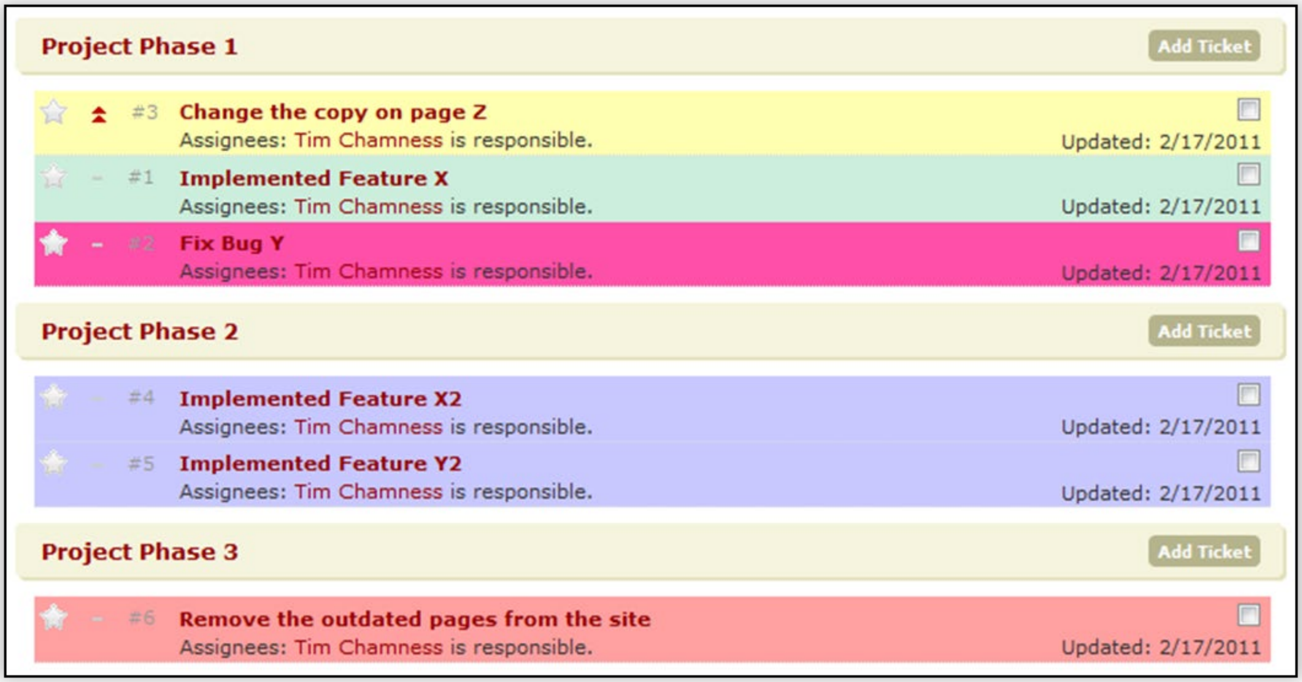


Active milestones

Dashboard » Projects » ACME Social Network » Milestones » Active

☆ -	Project Phase 1	Feb 22. 2011 — Feb 28. 2011	Due in 11 Days
☆ -	Project Phase 2	Mar 01. 2011 — Mar 07. 2011	Due in 18 Days
☆ -	Project Phase 3	Mar 08. 2011 — Mar 14. 2011	Due in 25 Days

Tickets: The project manager then breaks each iteration into smaller development tasks.



Project Phase 1 — Add Ticket

#3 **Change the copy on page Z** □
Assignees: Tim Chamness is responsible. Updated: 2/17/2011

#1 **Implemented Feature X** □
Assignees: Tim Chamness is responsible. Updated: 2/17/2011

#2 **Fix Bug Y** □
Assignees: Tim Chamness is responsible. Updated: 2/17/2011

Project Phase 2 — Add Ticket

#4 **Implemented Feature X2** □
Assignees: Tim Chamness is responsible. Updated: 2/17/2011

#5 **Implemented Feature Y2** □
Assignees: Tim Chamness is responsible. Updated: 2/17/2011

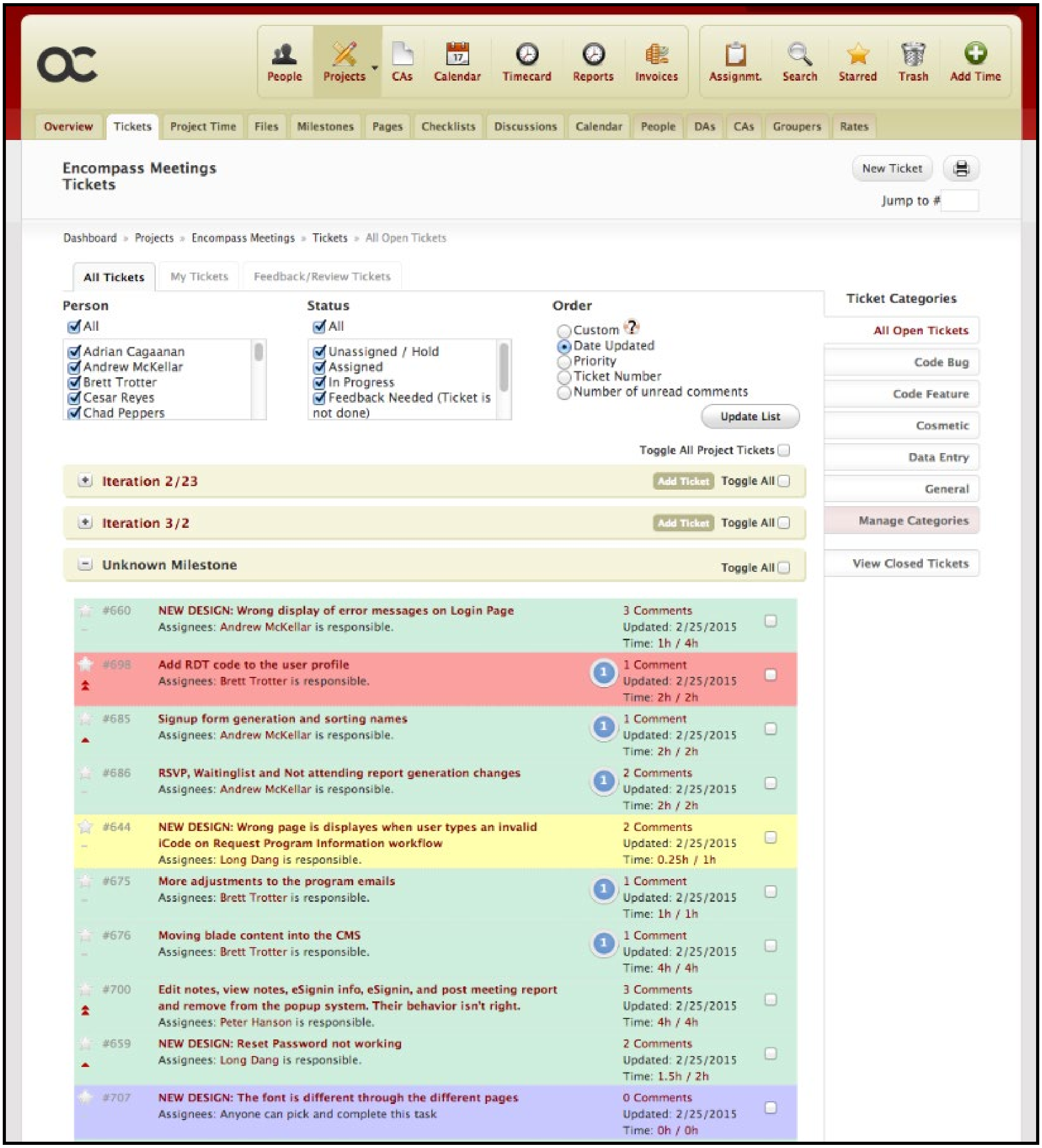
Project Phase 3 — Add Ticket

#6 **Remove the outdated pages from the site** □
Assignees: Tim Chamness is responsible. Updated: 2/17/2011

The project manager assigns each ticket to the most appropriate developer:

- If the ticket relates to existing code (like a bug fix) the ticket will be assigned to the person who knows it best.
- If the ticket involves specialized work it will be assigned to a person who is proficient in that technology.
- Otherwise the ticket will be assigned to the most available developer.

Tickets – Global View: This shows all of the tickets and the search parameters available to present the desired set of tickets. There are also tabs to work on just your tickets, as well as communicate on tickets that need it.



Encompass Meetings
Tickets

Dashboard » Projects » Encompass Meetings » Tickets » All Open Tickets

Person
☑ All
☑ Adrian Cagaanan
☑ Andrew McKellar
☑ Brett Trotter
☑ Cesar Reyes
☑ Chad Peppers

Status
☑ All
☑ Unassigned / Hold
☑ Assigned
☑ In Progress
☑ Feedback Needed (Ticket is not done)

Order
○ Custom ?
● Date Updated
○ Priority
○ Ticket Number
○ Number of unread comments

Toggle All Project Tickets □

Iteration 2/23 — Toggle All □

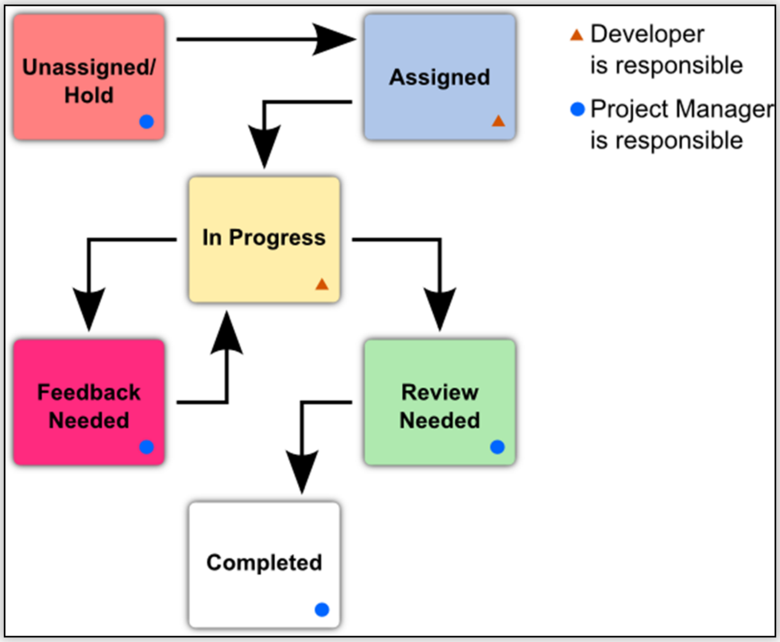
Iteration 3/2 — Toggle All □

Unknown Milestone — Toggle All □

#	Ticket	Status
#660	NEW DESIGN: Wrong display of error messages on Login Page Assignees: Andrew McKellar is responsible.	3 Comments Updated: 2/25/2015 Time: 1h / 4h
#698	Add RDT code to the user profile Assignees: Brett Trotter is responsible.	1 Comment Updated: 2/25/2015 Time: 2h / 2h
#685	Signup form generation and sorting names Assignees: Andrew McKellar is responsible.	1 Comment Updated: 2/25/2015 Time: 2h / 2h
#686	RSVP, Waitinglist and Not attending report generation changes Assignees: Andrew McKellar is responsible.	2 Comments Updated: 2/25/2015 Time: 2h / 2h
#644	NEW DESIGN: Wrong page is displayes when user types an invalid iCode on Request Program Information workflow Assignees: Long Dang is responsible.	2 Comments Updated: 2/25/2015 Time: 0.25h / 1h
#675	More adjustments to the program emails Assignees: Brett Trotter is responsible.	1 Comment Updated: 2/25/2015 Time: 1h / 1h
#676	Moving blade content into the CMS Assignees: Brett Trotter is responsible.	1 Comment Updated: 2/25/2015 Time: 4h / 4h
#700	Edit notes, view notes, eSignin info, eSignin, and post meeting report and remove from the popup system. Their behavior isn't right. Assignees: Peter Hanson is responsible.	3 Comments Updated: 2/25/2015 Time: 4h / 4h
#659	NEW DESIGN: Reset Password not working Assignees: Long Dang is responsible.	2 Comments Updated: 2/25/2015 Time: 1.5h / 2h
#707	NEW DESIGN: The font is different through the different pages Assignees: Anyone can pick and complete this task	0 Comments Updated: 2/25/2015 Time: 0h / 0h

Ticket Categories
- All Open Tickets
- Code Bug
- Code Feature
- Cosmetic
- Data Entry
- General
- Manage Categories
- View Closed Tickets

Ticket Statuses: This is the progression of a ticket.

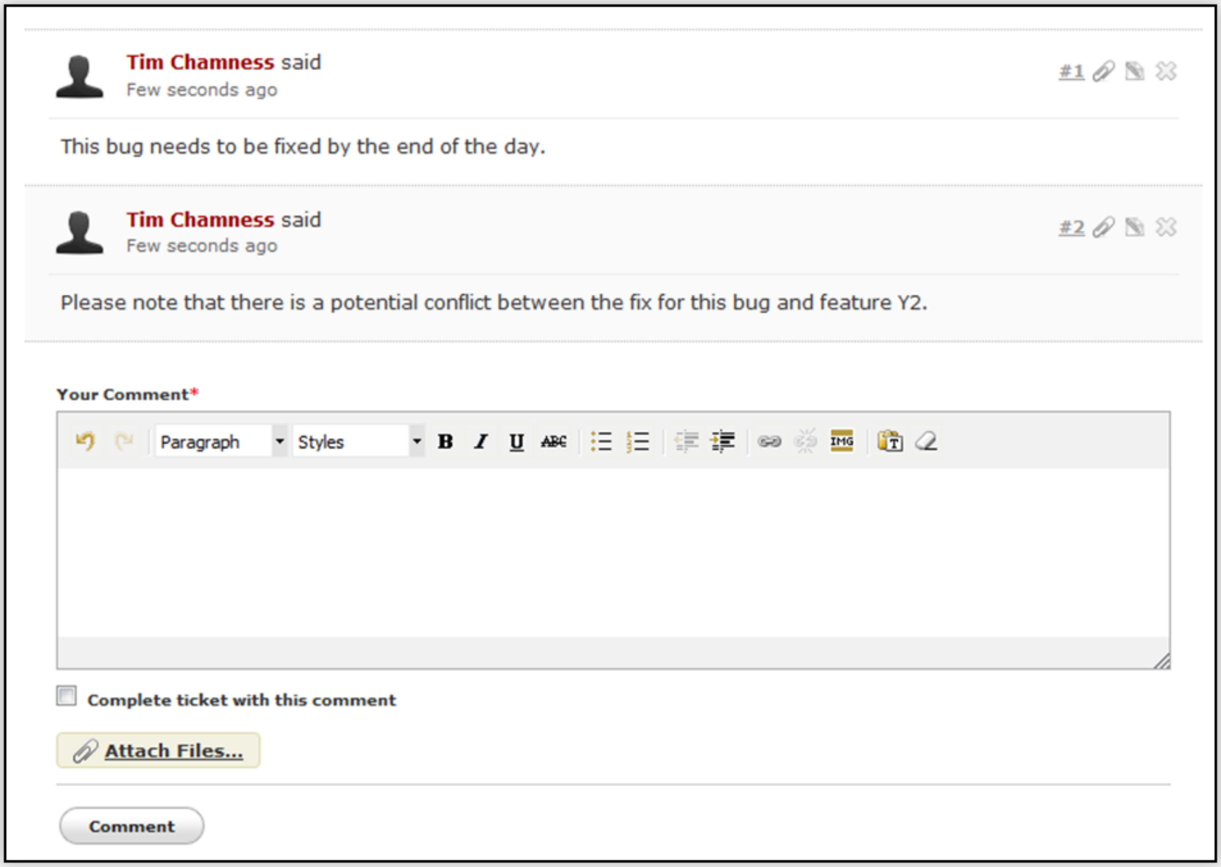


- Every ticket has a status. Ticket Status Descriptions:
  - Unassigned / Hold - work on this ticket should not proceed
  - Assigned - work on this ticket has not been started yet, but can be started at any time
  - In Progress - work on this ticket is partially completed
  - Feedback Needed - the ticket’s developer has a question; work is stopped for the moment
  - Review Needed - work on the ticket is complete, and needs to be reviewed by the project manager
  - Complete - the project manager has reviewed this, deemed it is internally complete, and will request a client review
- The ticket follows a defined workflow from conception to completion.
- The status indicates who is responsible for the ticket.

Ticket Workflow Advantages:

- Communication: it is always clear who must take the next steps on the ticket. This helps ensure that development does not stall due to a miscommunication.
- Efficiency: the “In Progress” status tells a project manager when a developer has already started working on a ticket. The project manager then knows that they should not re-assign the ticket. This improves efficiency by preventing two developers from working on the same task.
- Quality: the work done for every ticket is reviewed by at least two people: the developer and the project manager. In some cases, something we recommend if the project supports it, there are dedicated QA team members involved too. The “Review Needed” step ensures that no ticket slips through the review process.
- Agility: if work on a ticket ever needs to stop, the project manager is able to halt it quickly by changing the ticket status to Hold. This prevents extra costs from being incurred from unneeded work.

Ticket Communication: Comments can be posted on every ticket to facilitate communication between the developer, project manager and client.



Tim Chamness said
Few seconds ago

#1

This bug needs to be fixed by the end of the day.

Tim Chamness said
Few seconds ago

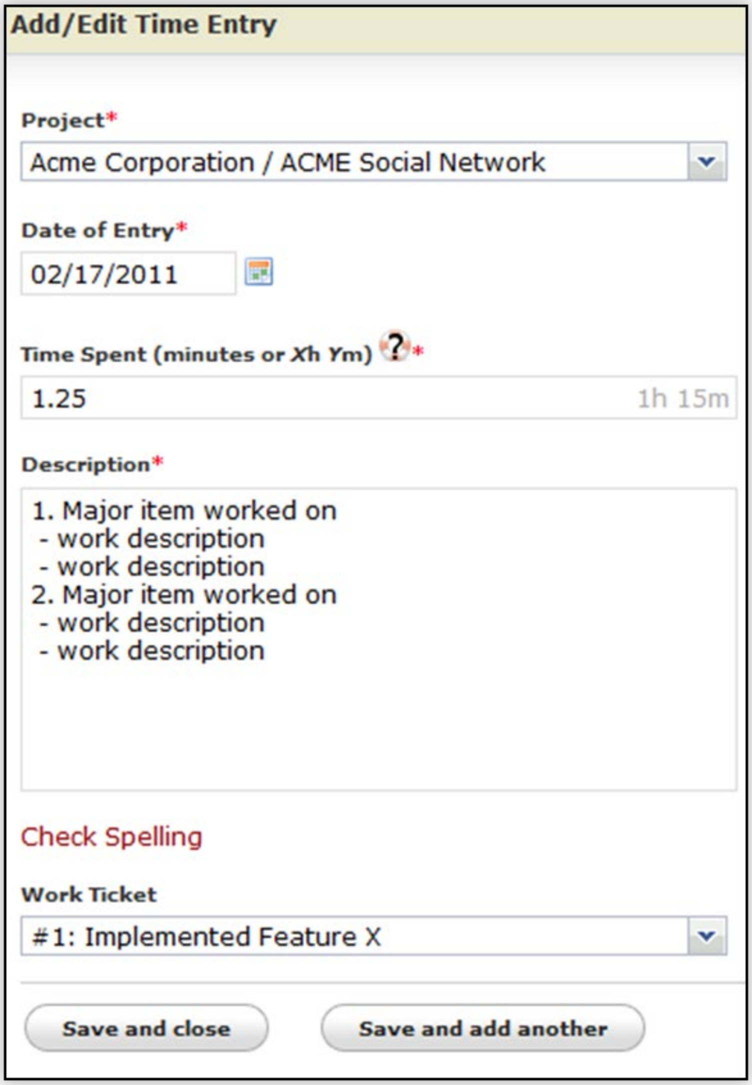
#2

Please note that there is a potential conflict between the fix for this bug and feature Y2.

Your Comment*

□ Complete ticket with this comment

Time Tracking: We enter in detailed time entries by task for both the client and our project manager. This time detail is sent out weekly to the client, and it's also sent again with the related invoice.



Add/Edit Time Entry

Project*
Acme Corporation / ACME Social Network

Date of Entry*
02/17/2011

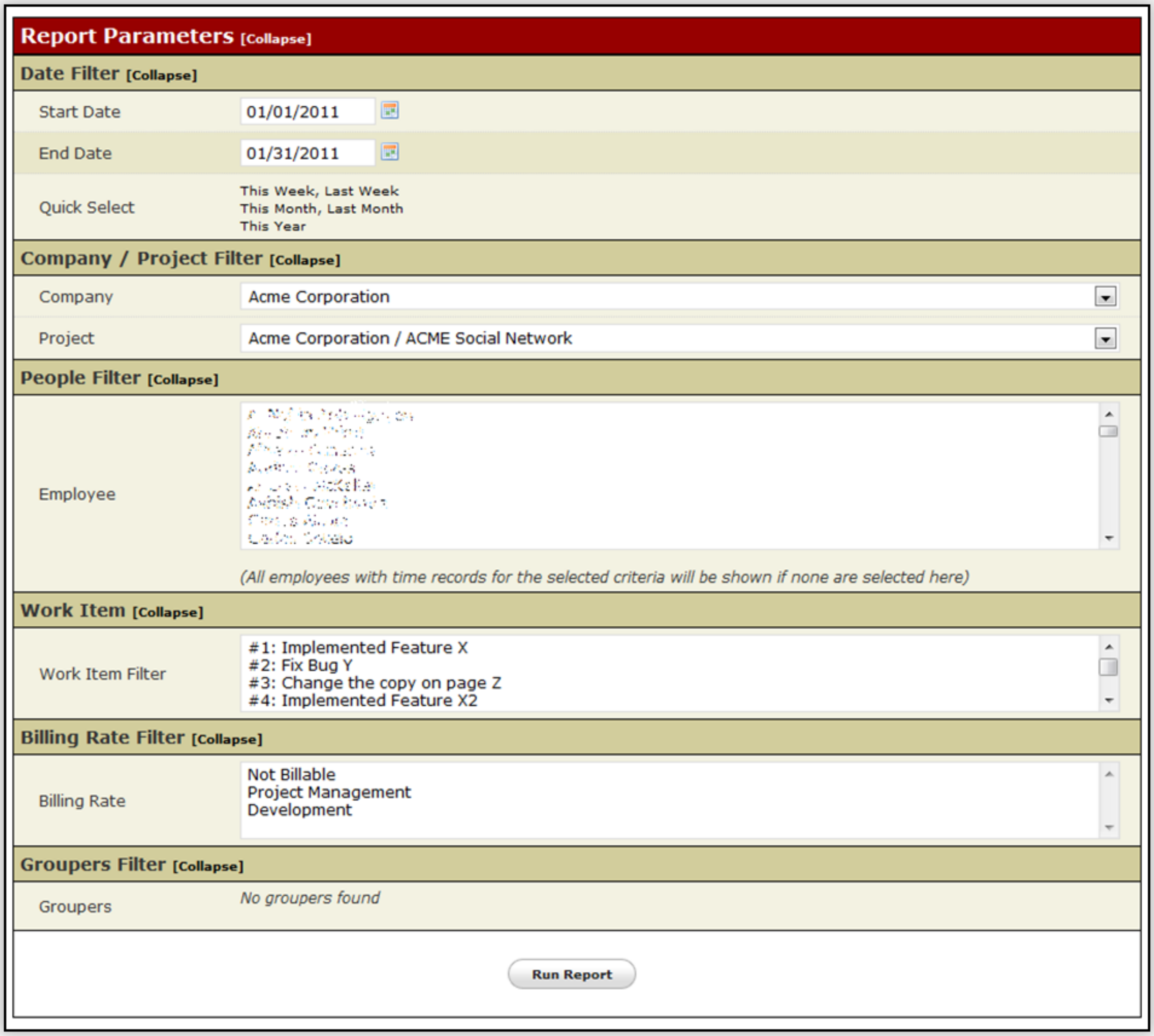
Time Spent (minutes or Xh Ym) ?*
1.25 1h 15m

Description*
1. Major item worked on
 - work description
 - work description
2. Major item worked on
 - work description
 - work description

Work Ticket
#1: Implemented Feature X

- Developers and project managers create time entries for every piece of work they do.
- Every time entry is associated with a ticket, allowing for detailed analysis of exactly how time is spent.

Time Reporting: Up and Running's custom time report allows for detailed analysis and review of time spent. Work can be categorized or "tagged" many ways, allowing for multiple views into the time.



Report Parameters [Collapse]

Date Filter [Collapse]

Start Date	01/01/2011
End Date	01/31/2011
Quick Select	This Week, Last Week This Month, Last Month This Year

Company / Project Filter [Collapse]

Company	Acme Corporation
Project	Acme Corporation / ACME Social Network

People Filter [Collapse]

Employee	[redacted] *(All employees with time records for the selected criteria will be shown if none are selected here)*

Work Item [Collapse]

Work Item Filter	#1: Implemented Feature X #2: Fix Bug Y #3: Change the copy on page Z #4: Implemented Feature X2

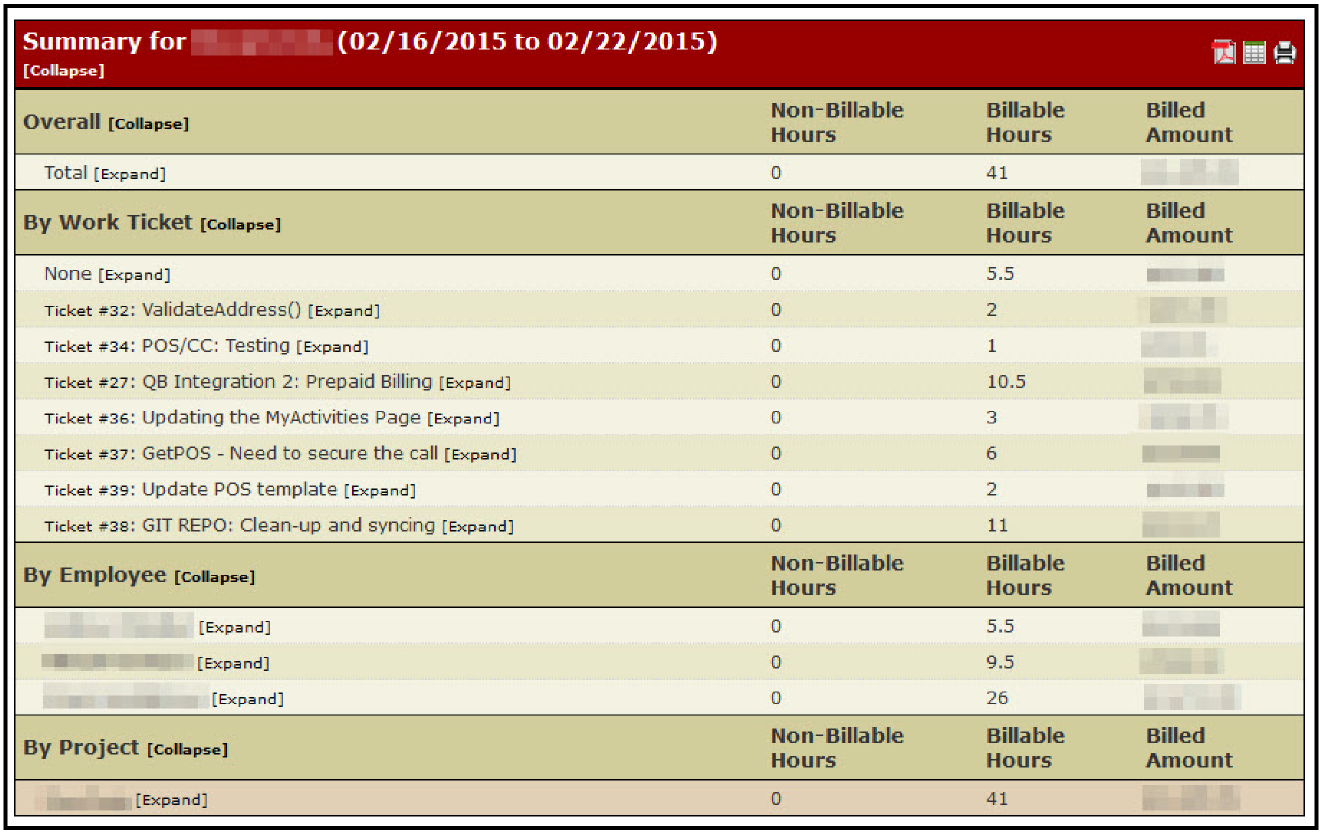
Billing Rate Filter [Collapse]

Billing Rate	Not Billable Project Management Development

Groupers Filter [Collapse]

Groupers	*No groupers found*

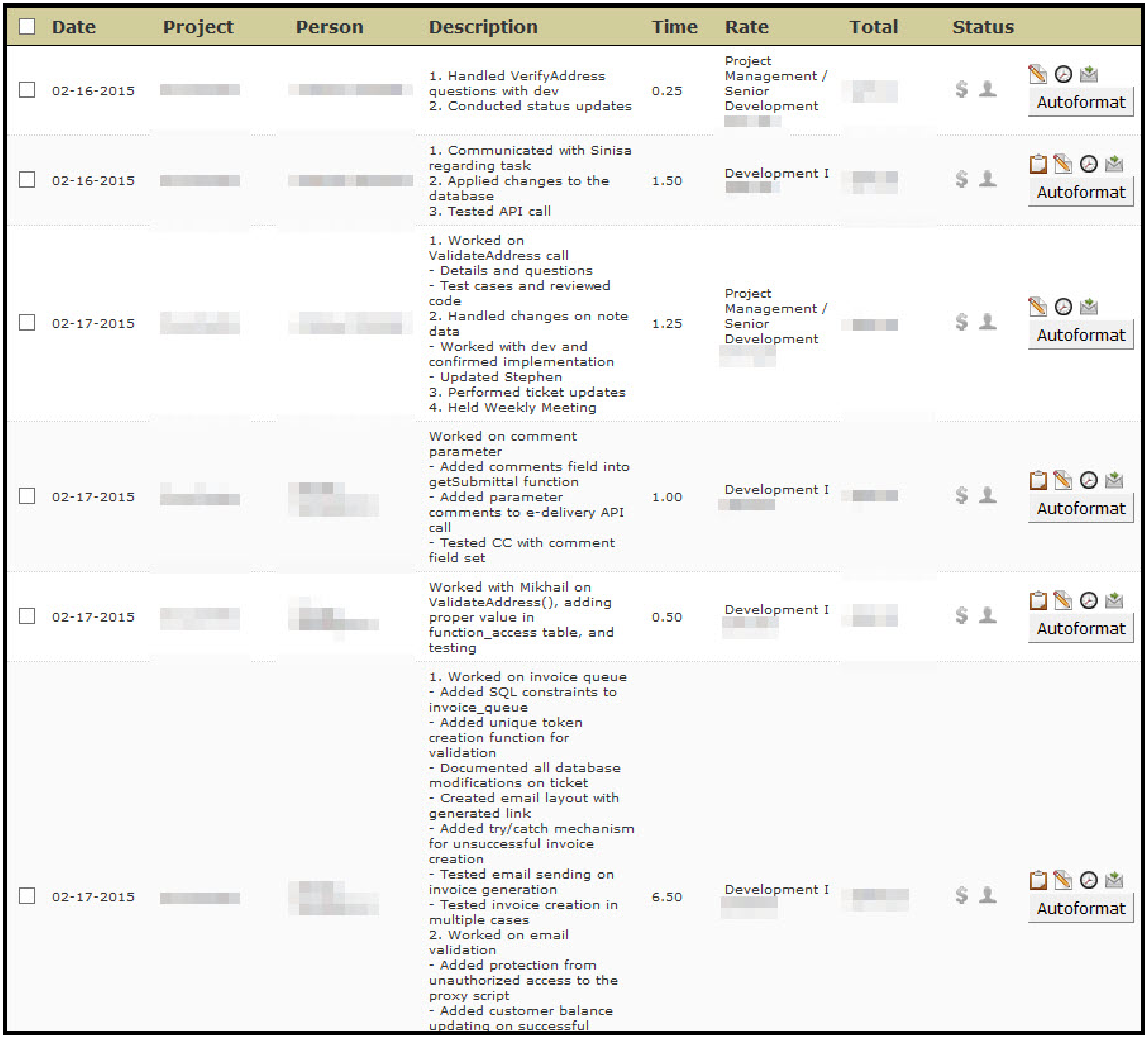
Time Reporting - Continued: The summary section can give a quick overview of the project status.



Summary for [redacted] (02/16/2015 to 02/22/2015)
[Collapse]

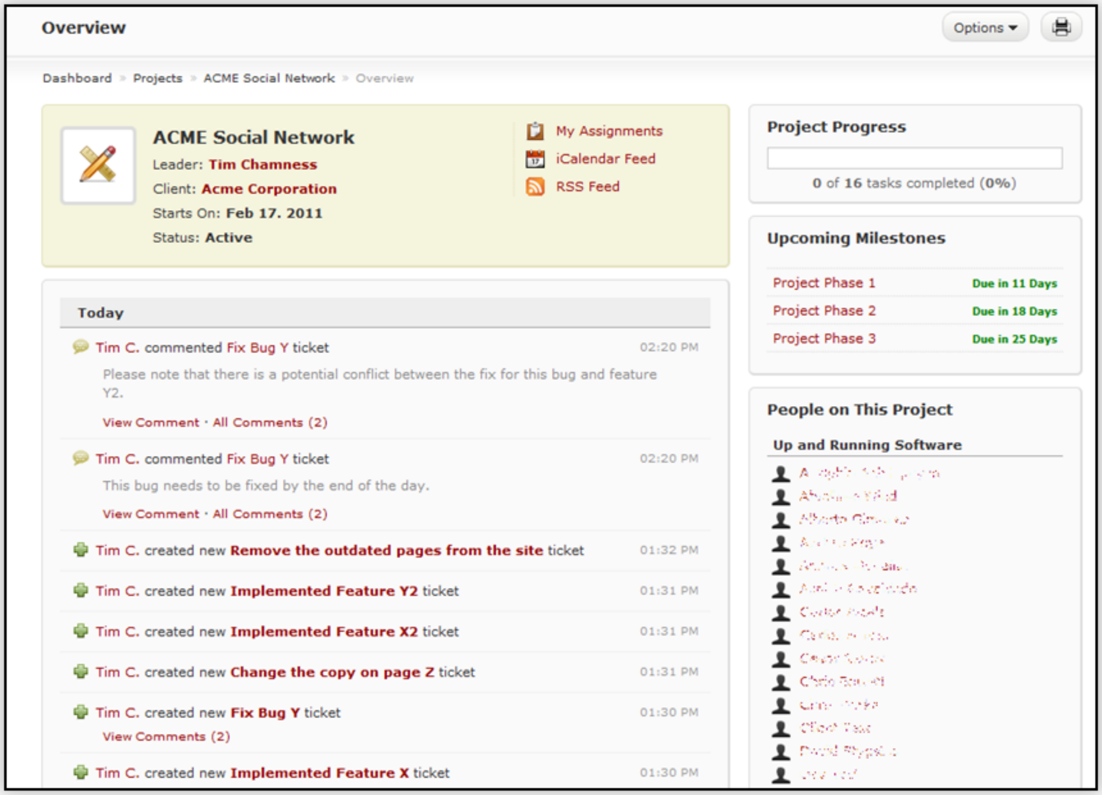
Overall [Collapse]	**Non-Billable Hours**	**Billable Hours**	**Billed Amount**
Total [Expand]	0	41	[redacted]
By Work Ticket [Collapse]	**Non-Billable Hours**	**Billable Hours**	**Billed Amount**
None [Expand]	0	5.5	[redacted]
Ticket #32: ValidateAddress() [Expand]	0	2	[redacted]
Ticket #34: POS/CC: Testing [Expand]	0	1	[redacted]
Ticket #27: QB Integration 2: Prepaid Billing [Expand]	0	10.5	[redacted]
Ticket #36: Updating the MyActivities Page [Expand]	0	3	[redacted]
Ticket #37: GetPOS - Need to secure the call [Expand]	0	6	[redacted]
Ticket #39: Update POS template [Expand]	0	2	[redacted]
Ticket #38: GIT REPO: Clean-up and syncing [Expand]	0	11	[redacted]
By Employee [Collapse]	**Non-Billable Hours**	**Billable Hours**	**Billed Amount**
[redacted] [Expand]	0	5.5	[redacted]
[redacted] [Expand]	0	9.5	[redacted]
[redacted] [Expand]	0	26	[redacted]
By Project [Collapse]	**Non-Billable Hours**	**Billable Hours**	**Billed Amount**
[redacted] [Expand]	0	41	[redacted]

Time Reporting - Continued: The details section provides detailed information on every piece of work done.



☐	Date	Project	Person	Description	Time	Rate	Total	Status	
☐	02-16-2015	[redacted]	[redacted]	1. Handled VerifyAddress questions with dev 2. Conducted status updates	0.25	Project Management / Senior Development [redacted]	[redacted]	$	Autoformat
☐	02-16-2015	[redacted]	[redacted]	1. Communicated with Sinisa regarding task 2. Applied changes to the database 3. Tested API call	1.50	Development I [redacted]	[redacted]	$	Autoformat
☐	02-17-2015	[redacted]	[redacted]	1. Worked on ValidateAddress call - Details and questions - Test cases and reviewed code 2. Handled changes on note data - Worked with dev and confirmed implementation - Updated Stephen 3. Performed ticket updates 4. Held Weekly Meeting	1.25	Project Management / Senior Development [redacted]	[redacted]	$	Autoformat
☐	02-17-2015	[redacted]	[redacted]	Worked on comment parameter - Added comments field into getSubmittal function - Added parameter comments to e-delivery API call - Tested CC with comment field set	1.00	Development I [redacted]	[redacted]	$	Autoformat
☐	02-17-2015	[redacted]	[redacted]	Worked with Mikhail on ValidateAddress(), adding proper value in function_access table, and testing	0.50	Development I [redacted]	[redacted]	$	Autoformat
☐	02-17-2015	[redacted]	[redacted]	1. Worked on invoice queue - Added SQL constraints to invoice_queue - Added unique token creation function for validation - Documented all database modifications on ticket - Created email layout with generated link - Added try/catch mechanism for unsuccessful invoice creation - Tested email sending on invoice generation - Tested invoice creation in multiple cases 2. Worked on email validation - Added protection from unauthorized access to the proxy script - Added customer balance updating on successful	6.50	Development I [redacted]	[redacted]	$	Autoformat

Project Dashboard: The system also provides a comprehensive project dashboard showing the project status and recent activities.



Overview

Options ▾

Dashboard » Projects » ACME Social Network » Overview

ACME Social Network
Leader: **Tim Chamness**
Client: **Acme Corporation**
Starts On: **Feb 17. 2011**
Status: **Active**

My Assignments
iCalendar Feed
RSS Feed

Project Progress

0 of **16** tasks completed (**0%**)

Upcoming Milestones

Project Phase 1	Due in 11 Days
Project Phase 2	Due in 18 Days
Project Phase 3	Due in 25 Days

People on This Project

Up and Running Software

[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]

Today

Tim C. commented Fix Bug Y ticket 02:20 PM
Please note that there is a potential conflict between the fix for this bug and feature Y2.
View Comment · All Comments (2)

Tim C. commented Fix Bug Y ticket 02:20 PM
This bug needs to be fixed by the end of the day.
View Comment · All Comments (2)

Tim C. created new **Remove the outdated pages from the site** ticket 01:32 PM

Tim C. created new **Implemented Feature Y2** ticket 01:31 PM

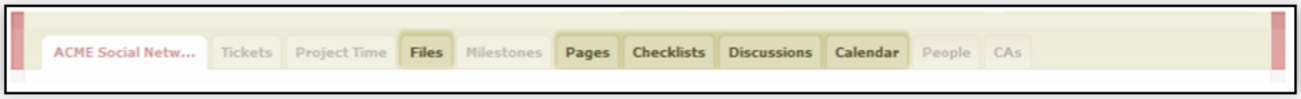
Tim C. created new **Implemented Feature X2** ticket 01:31 PM

Tim C. created new **Change the copy on page Z** ticket 01:31 PM

Tim C. created new **Fix Bug Y** ticket 01:30 PM
View Comments (2)

Tim C. created new **Implemented Feature X** ticket 01:30 PM

Other Collaboration Features: activeCollab provides numerous other collaboration tools:



- File Management for sharing specification documents
- Pages for sharing text-based project information
- Checklists for tracking specific work procedures
- Discussions for collaborating on topics of any sort
- A Calendar for tracking project deadlines

Demonstration

- We'd be happy to have a GoToMeeting session with you if you'd like.
- We can also prepare our system with specific data based on your development needs for the demonstration. We'd just need some example development tasks from you to do this.

OTHER PROCESS GUIDES:

We maintain two other process guides if you'd like to see them:

- **The Development Process (https://bit.ly/2U0dey8):** This presents some concepts to determine what's important in a software development project, as well as how to approach software development.

  In relation to this process guide, this document covers in some detail how tickets might be prioritized based on what the stakeholders value (the relationship of money, scope, quality, and time usually) and created based on the specifications-definition process by means of using mockups.

- **Discovery Process Methods (https://bit.ly/2Q4YkYP):** This document was written for people who'd like to create software and would like to be introduced to some methods that might help them with that, and it presents many discovery methods.

  In relation to this process guide, this document covers at a high level the source specifications that would be used to create the tickets.

NEXT STEPS:

I hope the information provided above is useful and that we can explore working together further. If you have questions, requests, or feedback, we'd be pleased to hear from you at any time.

Thank you and Respectfully,

Ian McKilligan

Ian McKilligan, CEO
Up and Running Software, Inc.
Cell: 906-281-2627
Email: ian@upandrunningsoftware.com