CRU App:
Quality Assurance Plan

Will Code for Food
Computer Science Department
California Polytechnic State University
San Luis Obispo, CA USA
December 2, 2015

Contents

Credits

Name	**Date**	**Role**	**Version**
Kayla Carr	11/28/2015	AC-4, AC-5, ST-2	1.0
Shelli Crispen	11/29/2015	AC-7,	1.0
Mallika Potter	11/30/2015	2.2.1, 2.2.2, AC-7, NF-3	1.0
Gavin Scott	11/22/2015	1, 2.1.5, 2.2.3, 3, 4.1 AC-1, 4.3 NF-1	1.0
Mason Stevenson	11/23/2015	AC-3, NF-2	1.0
Brian Quezada	11/27/2015	2.1.1, 2.1.2, 2.2.1, 2.1.3, AC-2, ST-1	1.0

1 Introduction

This document outlines the approach we will take towards ensuring the quality of our code. It will explain our testing, code style, and code-pushing practices, as well as who is responsible for ensuring that all new code is well tested. Important specific test cases, testing requirements previously outlined in our Software Requirements Specification, will be recorded here and implemented as the code is written. Any other details regarding the quality ensurance of our code and the app, such as what tools will be used, are also defined in this document.

2 Approach

2.1 Process

2.1.1 Committing

Every time code is committed, there must be unit tests committed with it. When the tests are created will be left up for the individual to decide, but it must be completed by the time the code is committed. All tests must pass. If UI is altered and already tested, it is the responsibility of the person changing the feature to update the tests and make sure they pass. If other tests fail in Bamboo or in Android Studio after a person has committed changes, it is the responsibility of the person who broke the build to fix it.

2.1.2 Code Reviews

All submitted code will undergo code review from at least one other person before it is pulled into the master branch. That person will ensure that the code is readable, reasonable, and correct. That person will also verify that tests were made for the submitted code.

2.1.3 Comments

Javadoc descriptions are required for classes and public and protected methods, except for overrides, getters, and setters. The description should make it clear what the method or class is doing and how it should be used. Descriptions for private methods are highly encouraged. Parameter and return descriptions are not required. The individual can feel free to add any more comments that they feel is necessary throughout their code.

2.1.4 Style Guidelines

We shall adhere to the coding style guidelines laid out by Cru. These can be found at the following link: Cru Java Guidelines.

2.1.5 Code Coverage

Code will be considered well-tested when it is accompanied by a suite of JUnit tests, all of which pass and bring the code coverage of the new code to at least 80%.

2.2 Tools

2.2.1 Code Reviews

We will use pull requests within Bitbucket to undergo code reviews. When beginning new code or a new task, a new branch should be made. This can be done by selecting "create branch" on a ticket in Jira, or by selecting "create branch" in Bitbucket. When the work is complete, the code must be pushed to the remote branch, and then a pull request must be created in Bitbucket. A reviewer must be chosen to review the code. Once the code has been approved, the branch can be merged into the master branch.

2.2.2 Testing

We will use the Espresso Testing Framework for UI testing. UI testing will be due for non-temporary features on commit. Non-UI components will be tested using JUnit tests. Bamboo will run all of the JUnit tests every time a commit is pushed through.

2.2.3 Code Coverage

Code coverage will be measured using the Gradle code-coverage tool built into Android Studio.

3 Key Participants

There is no one member of the team more responsible for testing than any other member. Each member is expected to only commit code that has been tested and code-reviewed by another member of the team. All tests will be run when new code is pushed, and if a new change breaks any existing tests it is the responsibility of the person pushing the code to address the problem.

4 Test Cases

4.1 Acceptance Criteria

Test Case ID:	AC - 1
Test Case Name:	Enroll as a driver
Requirement:	SR 4.6.1, 4.6.2, and 4.6.4
Created By:	Gavin Scott
Last Updated By:	Gavin Scott

Date Created:	November 20, 2015	
Date Last Updated:	November 20, 2015	
Preconditions:	The user is verified within the database as a valid driver.	
Actions:	**Step**	**Expected Output**
	User selects the ride sharing tab.	System displays a list of events with ridesharing enabled.
	User selects an event.	System displays the option to either drive or ride for the event.
	User selects that they can drive.	System displays a form prompting the user to enter the date and time they wish to leave, where they would like to leave from, how many seats are available, and whether or not they are driving two-ways.
	User enters valid input and submits the form.	System sends the information to the database and displays a success message. System returns to the event page. The option to sign up as a driver has been replaced by the option to view their current ride.

Test Case ID:	AC - 2	
Test Case Name:	Join a Community Group	
Requirement:	SRS 4.5	
Created By:	Brian Quezada	
Last Updated By:	Brian Quezada	
Date Created:	November 22, 2015	
Date Last Updated:	November 22, 2015	
Preconditions:	User has joined at least one ministry	
Actions:	**Step**	**Expected Output**
	User goes to the Join Community Group section of Get Involved	System displays a list of ministries that they have joined.
	User selected a ministry	System displays questions such as gender, availability,

		and name.
	User submits valid answers	System sends the information to the database and displays the information of matching community groups.
	User selects a community group to join.	The application displays the information of a leader of the group. The community group leader receives the name and number of the member through a text.

Test Case ID:	AC - 3	
Test Case Name:	Access Ministry Resources	
Requirement:	SRS 4.2	
Created By:	Mason Stevenson	
Last Updated By:	Mason Stevenson	
Date Created:	November 23, 2015	
Date Last Updated:	November 23, 2015	
Preconditions:	none	
Actions:	**Step**	**Expected Output**
	User selects the resources tab.	System displays available resource categories: articles, videos, leader articles
	User selects articles category.	System displays list of available articles by date.
	User selects an article.	System displays the corresponding article.
	User selects go back arrow.	System returns to resource category section.
	User selects videos category.	System displays list of available videos by date.
	User selects a video.	System prompts user to open video with YouTube or browser.

Test Case ID:	AC - 4	
Test Case Name:	Join a Summer Mission	
Requirement:	SRS 4.9	
Created By:	Kayla Carr	
Last Updated By:	Kayla Carr	
Date Created:	November 28, 2015	
Date Last Updated:	November 28, 2015	
Preconditions:	User has joined at least one ministry and campus	
Actions:	**Step**	**Expected Output**
	User selects the summer missions tab.	System displays a list of mission trips available at the user's campus.
	User selects a mission trip.	System retrieves and displays information about the trip, such as a description, length, cost, leadership, and location.
	User selects apply.	System prompts user to open up a link to the webpage where the user can apply for the mission on a browser of the user's choice.

Test Case ID:	AC - 5	
Test Case Name:	View Ministry Events	
Requirement:	SRS 4.8	
Created By:	Kayla Carr	
Last Updated By:	Kayla Carr	
Date Created:	November 28, 2015	
Date Last Updated:	November 28, 2015	
Preconditions:	User has joined at least one ministry and campus	
Actions:	**Step**	**Expected Output**
	User selects the events	System displays a list of upcoming events for the user's

	tab.	subscribed ministries and campuses, ordered by their upcoming date.
	User selects an event from the list of events.	System retrieves and displays the date, description, location, and time of the event, and whether rides are available.
	User selects add to calendar on the event.	System creates a calendar event for the event and adds it to the user's native Google Calendar app.
	User selects ride sharing option on the event, if it is available.	System opens up the ride sharing tab and displays the ride information for the chosen event.

Test Case ID:	AC - 6	
Test Case Name:	Subscribe To a Ministry	
Requirement:	SRS 4.1	
Created By:	Mallika Potter	
Last Updated By:	Mallika Potter	
Date Created:	November 29, 2015	
Date Last Updated:	November 30, 2015	
Preconditions:	The user has subscribed to a campus.	
Actions:	**Step**	**Expected Output**
	User enters the choose ministries page.	System displays a list of ministries affiliated with the user's campus.
	User selects at least one ministrie.	System sends the information to the database and displays a success message.

Test Case ID:	AC - 7
Test Case Name:	Cancel a Ride
Requirement:	SRS 3.15
Created By:	Shelli Crispen

Last Updated By:	Shelli Crispen	
Date Created:	November 29, 2015	
Date Last Updated:	November 29, 2015	
Preconditions:	Used ride sharing to get a ride to an event.	
Actions:	**Step**	**Expected Output**
	User selects the ride sharing tab.	System displays a list of events enabled for ride sharing and an option for your rides.
	User selects "Your Rides" option. User selects the ride they wish to delete.	System displays a list of all rides the user is signed up for. System displays the ride information and a add to calendar buttons as well as a delete button.
	User clicks the delete button.	System displays a success message and returns to the "Your Rides" page.

Test Case ID:	AC - 8	
Test Case Name:	Register as a Rider	
Requirement:	SRS 4.7	
Created By:	Shelli Crispen	
Last Updated By:	Shelli Crispen	
Date Created:	November 29, 2015	
Date Last Updated:	November 29, 2015	
Preconditions:	Be a CRU member	
Actions:	**Step**	**Expected Output**
	User selects the ride sharing tab.	System displays a list of events enabled for ride sharing and an option for your rides.
	User selects the event they want a ride to.	System displays a form prompting the user to enter the date and time they would like to be picked up and dropped off. They will also need to select if they are riding one way or both ways.

	User submits the form.	System displays the top ten rides that match the user's input.
	User selects a ride from the list	System adds the ride to the user's "Your Rides" section.

4.2 System Tests

Test Case ID:	ST-1	
Test Case Name:	Send a Push Notification	
Requirement:	SRS 4.3	
Created By:	Brian Quezada	
Last Updated By:	Brian Quezada	
Date Created:	November 22, 2015	
Date Last Updated:	November 22, 2015	
Preconditions:	User(s) has joined an event	
Actions:	**Step**	**Expected Output**
	Administrator accesses an account on Parse.com.	Our application is an available app.
	Administrator selects Send a Push, Custom Audience, Add a Condition	Android is an available platform. A channel is available for the desired event.
	Administrator creates a time and message for a notification and selects Send.	All users that have joined the event receive the push notification.

Test Case ID:	ST-2
Test Case Name:	Update Calendar Events
Requirement:	SRS 4.8

Created By:	Kayla Carr	
Last Updated By:	Kayla Carr	
Date Created:	November 28, 2015	
Date Last Updated:	November 28, 2015	
Preconditions:	User has added an event to their Google Calendar through the app.	
Actions:	**Step**	**Expected Output**
	Administrator accesses the Cru event database and changes the date of an upcoming event.	System is notified of the update in the database and checks whether or not the user has added said event to their calendar. If they have, the system sends a call to update the previously created Google Calendar event.

4.3 Non-functional Tests

Test Case ID:	NF - 1	
Test Case Name:	Fill Up Ride	
Requirement:	PR - 1	
Created By:	Gavin Scott	
Last Updated By:	Gavin Scott	
Date Created:	November 20, 2015	
Date Last Updated:	November 20, 2015	
Preconditions:	1. The user is viewing the list of available rides for an event. 2. There is a ride with only one available seat left.	
Actions:	**Step**	**Expected Output**
	User selects a ride with only one seat remaining.	System updates the ride in the database as "full" and displays a confirmation message.
	User navigates back to the list of available rides for the event and refreshes.	Within 10 seconds (and upon multiple refreshes) the ride the user signed up for will no longer be shown as available.

Test Case ID:	NF - 2	
Test Case Name:	Modularize the Application	
Requirement:	SQA - 1	
Created By:	Mason Stevenson	
Last Updated By:	Mason Stevenson	
Date Created:	November 23, 2015	
Date Last Updated:	November 23, 2015	
Preconditions:	none	
Actions:	**Step**	**Expected Output**
	Remove the Events model, and delete the link named "Events" from the app's main menu.	The app launches and all other sections available from the main menu except "Rides" still meet their acceptance criteria.
	Remove the Resources model, and delete the link named "Resources" from the app's main menu.	The app launches and all other sections available from the main menu still meet their acceptance criteria.
	Remove the SummerMissions model, and delete the link named "Summer Missions" from the app's main menu.	The app launches and all other sections available from the main menu still meet their acceptance criteria.
	Remove the GetInvolved model, and delete the link named "Get Involved" from the app's main menu.	The app launches and all other sections available from the main menu still meet their acceptance criteria.
	Remove the Rides model, and delete the link named "Ride Share" from the app's main menu.	The app launches and all other sections available from the main menu still meet their acceptance criteria.

Test Case ID:	NF - 3	
Test Case Name:	Driver Verification	
Requirement:	SaR-2	
Created By:	Mallika Potter	
Last Updated By:	Mallika Potter	
Date Created:	November 29, 2015	
Date Last Updated:	November 29, 2015	
Preconditions:	User has requested to be a driver for a specific event.	
Actions:	**Step**	**Expected Output**
	Application checks phone number associated with user's downloaded version of application against list of approved drivers.	Database contains user's phone number in approved driver's list.