Better Ballot

Software Architecture Document

Johnnie Cooper, Alex Ross, Soren Christiansen

Version 1.0

11/23/2018

Revision History

Version	Description of Versions / Changes	Responsible Party	Date
1.0	Initial version	Cooper, Soren, Alex	11/23/13

1. Table of Contents

Software Architecture Document

2. *Introduction/Vision*

This document provides a high level overview and explains the architecture of Better Ballot which is an electronic voting software system which will be used for the State of South Carolina midterm elections, but not limited to one specific governing body. Better Ballot is designed to be dynamic and versatile, to provide a configurable electronic voting solution for any scenario that is needed.

2.1. Purpose

The *Software Architecture Document* (SAD) provides a comprehensive architectural overview of the Better Ballot electronic voting system. This documents presents a number of different architectural views and diagrams to depict the different aspects of the systems functional and nonfunctional features.

2.2. Scope

The scope of this SAD is to explain the architecture of the Better Ballot electronic voting system. This document describes the various aspects of the electronic voting system design that the Stakeholders and users who require a technical understanding of the system. The architecture is based on the Unified Process.

2.3. Definitions, Acronyms, and Abbreviations

- **UP** – Unified Process
- **Better Ballot** – Software designed to provide a streamlined versatile reliable voting solution
- **Client** – The process running on a machine that a user will use to place votes
- **Voter** – The user which will be casting the vote to the ballot system
- **Administrator** – Individual tasked with the administrate, upkeep and oversight of the Better Ballot clients and their host machines
- **SAD -** Software Architecture Document
- **UML** – Unified Modeling Language
- **AWS** – Amazon Web Services
- **RDBMS –** Relational Database Management System
- **SSD –** System Sequence Diagram

2.4. References

None.

2.5. Overview

In order to fully document all the aspects of the architecture, the Software Architecture Document contains the following subsections.

Section 2: summary of the first two phases in the Unified Process which the system followed, and implemented in the development of the system.

Section 3: describes the architectural goals and constraints of the system

Section 4: describes the most important use-case diagrams

Section 5: describes logical view of the system including interface and operation definitions.

Section 6: describes significant persistence elements.

Section 7: describes how the system will be deployed.

3. Architectural Representation

Following the phases of the UP, this SAD will cover the summary development and design of phases of the UP.

Use Case view

Audience: all the stakeholders of the system, including the admin-users.
Area: describes the set of scenarios and/or use cases that represent some significant, central functionality of the system. Describes the actors and use cases for the system, this view presents the needs of the user and is elaborated further at the design level
Related Artifacts: Vison(See Introduction Section), Use-Case Model, Supplementary Specifications, and Risk List.

Logical View

Audience: Developers.
Area: Majority of the Functional Requirements: describes the design's object model. Also describes the most important use-case realizations and business requirements of the system.
Related Artifacts: Domain Model, Design model, Data Model, SSD, and Sequence Diagrams.

4. Architectural Specifications

There are some key requirements and system constraints that have a significant bearing on the architecture.

1. The system will be written using Oracles Java programming language with a JavaFX API but will use an open source RDBMS system for a central data storage platform which is also Oracles MySQL, and this will be deployed to an AWS RDBMS instance.

2. The system must be secure since it will be handling sensitive data.

4.1. Supplemental Specifications

1. Stable Internet Access is a requirement to connect to the RDBMS
2. Performance is a requirement for a quick accurate response to the user
3. Multiple User Compatibility to be able to hand multiple users connecting and voting at once.
4. Should be able to run on any OS that has Java installed on the machine
5. System should be handicap accessible and accommodate for any special needs.

4.2. Risk List

ID	Risk Description	Impact
1	Hardware	Possibly expensive hardware
2	Time	Delayed Results
3	Legal	Leaks of possible important info

4.3. Actors

Voter

The actor that would be using the Better Ballot system to cast a vote to a ballot that is created and configured by the administrator

Administrator

The actor that will be administrating the system. This actor is responsible for creating, configuring, and maintaining each of the ballots. This actor will also deliver the ballot results

5. *Use Case View*

The purpose of the inception phase is to get a common understanding of the background of the usage of the system and the interactions between its components which is agreed upon between the developers and product owners. This is a high level summary of the agreement of how the system will operate.

5.1. Use-Case Model

Use Case Scope can be viewed here

5.1.1. Use Cases

Fully Dress Use Case Documents: (Note: You may have to adjust hyperlinks in this document)

1 - Cast Vote

2 - Configure Ballot

3 - View Statistics

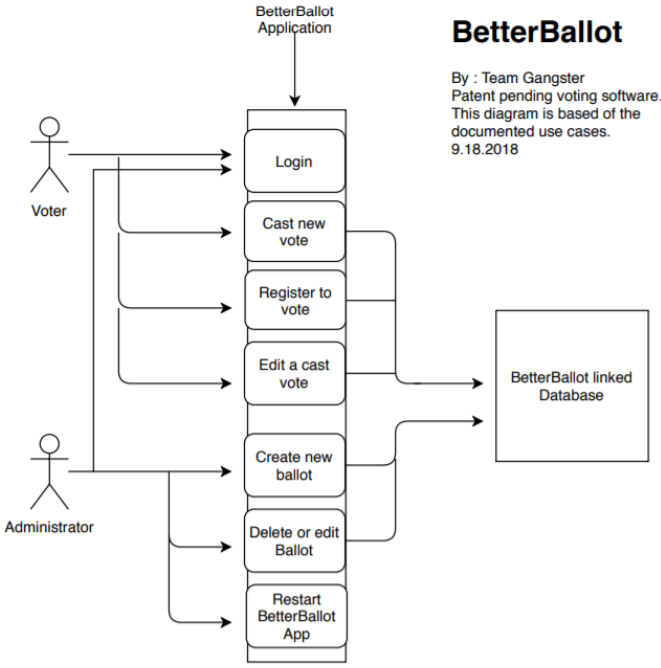

Figure 3.2 Use Cases Diagram

5.2. System Sequence Diagrams

5.2.1. Use Case – Cast Vote

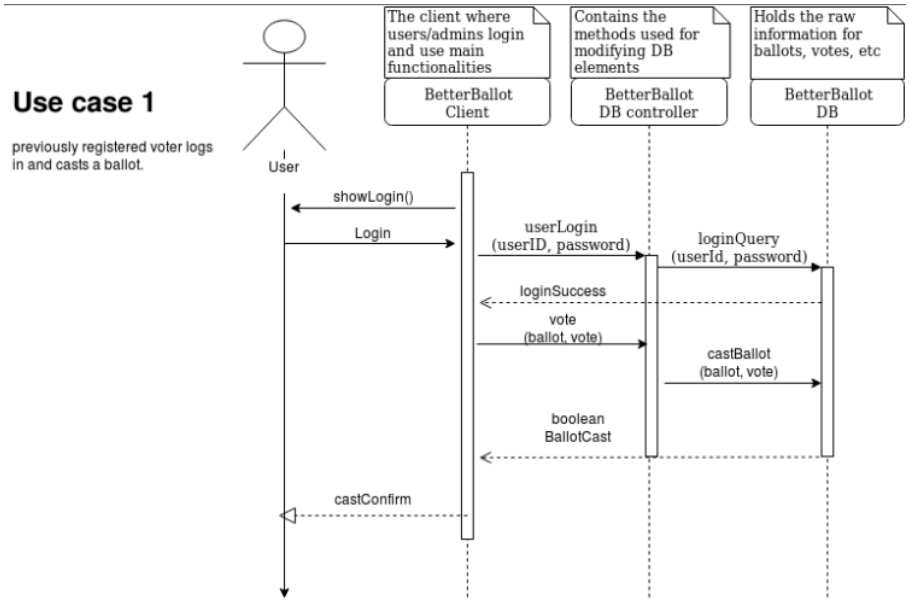

Figure 4.2.1 System Sequence Diagram – Use Case Cast Vote

5.2.2. Use Case – Register to Vote

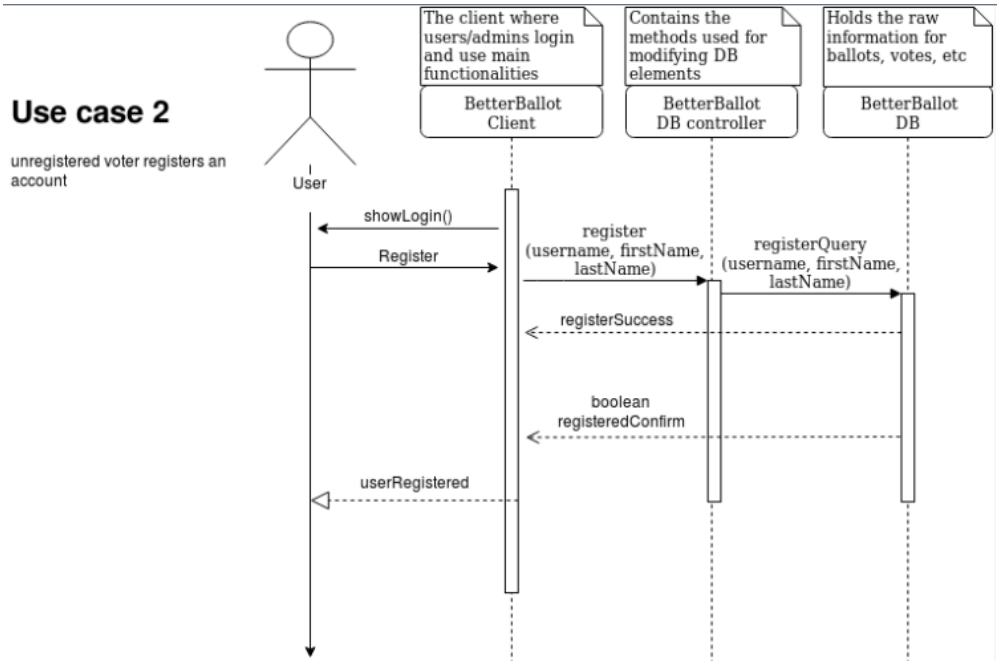

Figure 4.2.2 System Sequence Diagram – Use Case Register to Vote

5.2.3. Use Case – Admin Creates Ballot

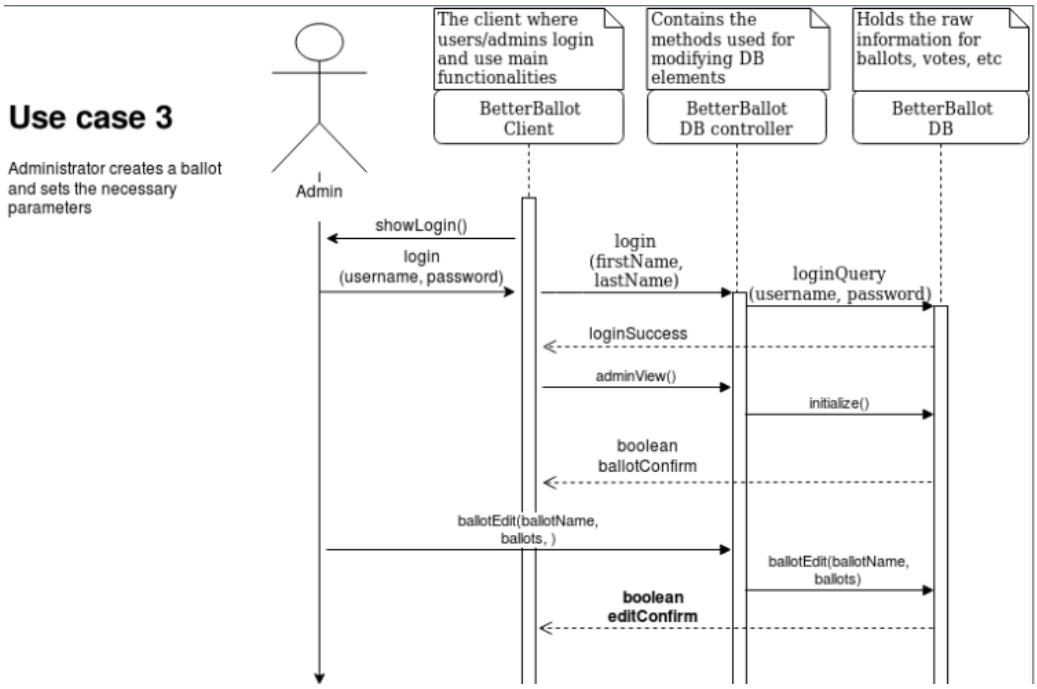

Figure 4.2.3 System Sequence Diagram – Use Case Create Ballot

5.2.4. Use Case – Admin Views Ballot Statistics

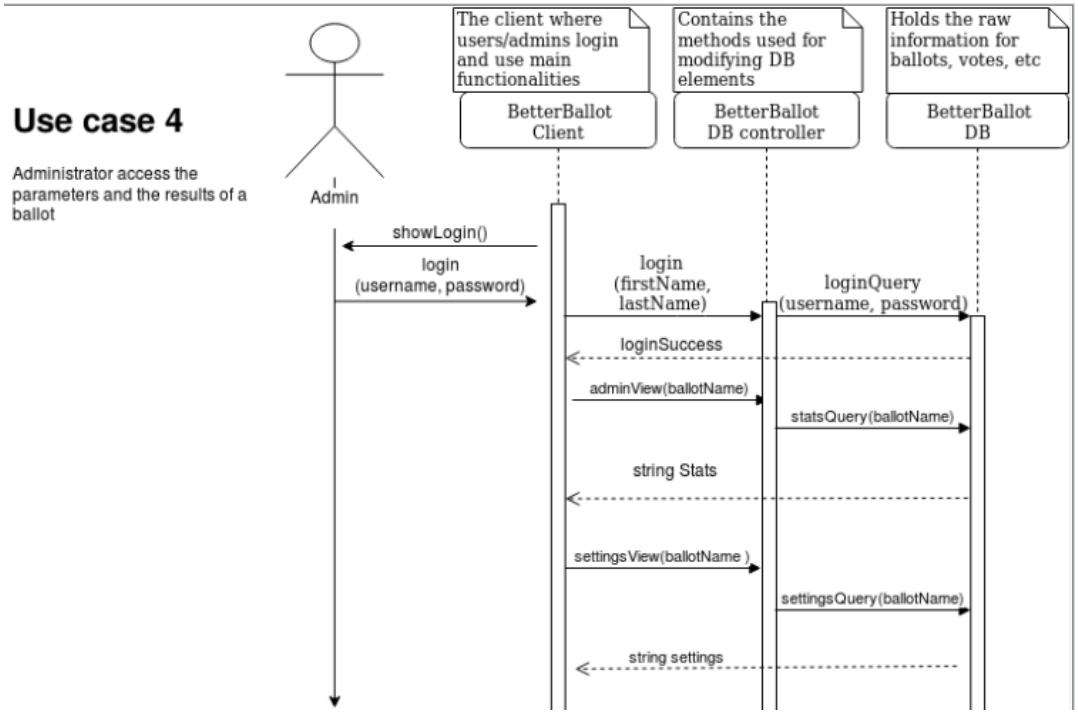

Figure 4.2.4 System Sequence Diagram – Use Case Admin Views Stats

6. *Logical View*

The main goal of the logical view is to define the components that will make up the system and to define the interfaces through which they will communicate and interact with one another.

6.1. *Domain Model*

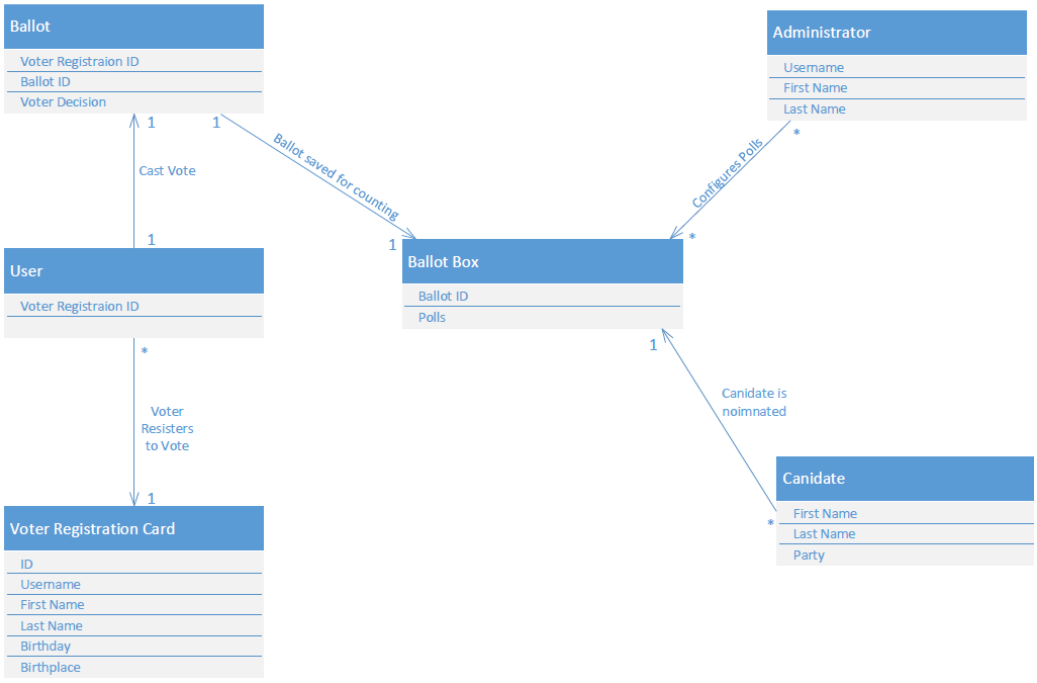

Figure 3.3 Domain Diagram

6.2. Class Diagram

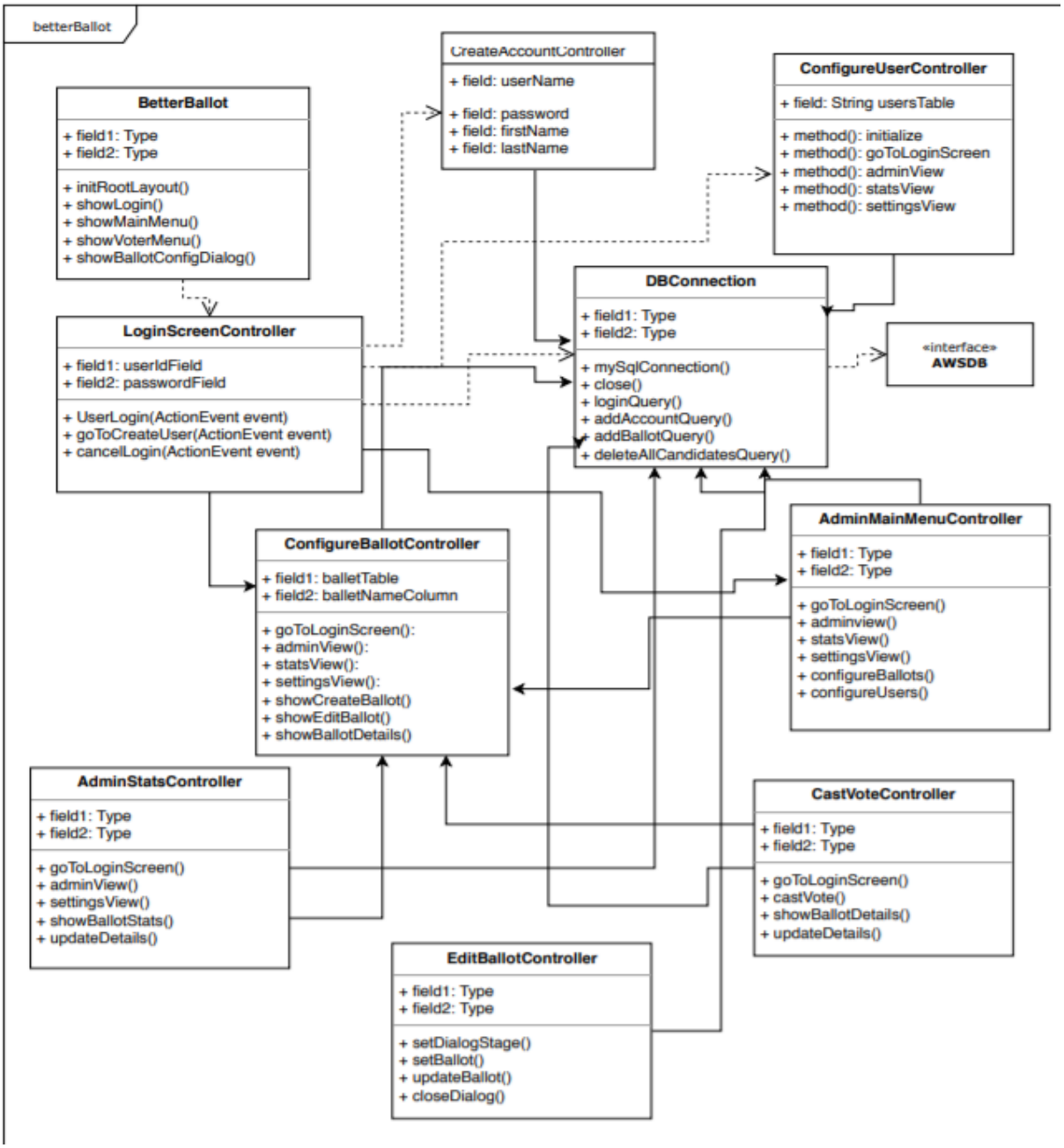

Figure 5.2 Class Diagram

6.3. Sequence Diagrams

6.3.1. Use Case Configure User

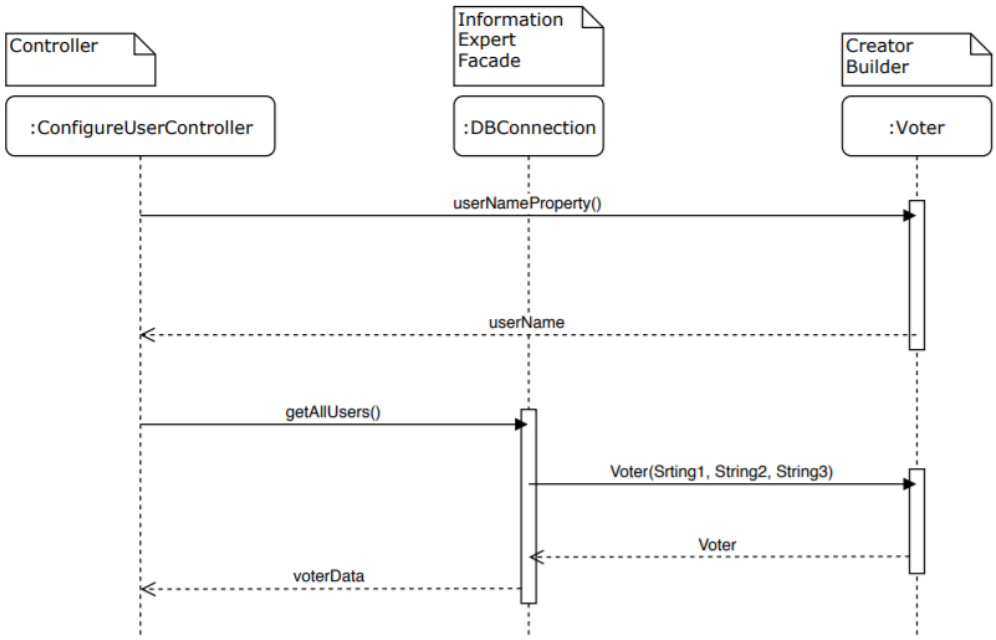

Figure 5.3.1 Sequence Diagram – Configure User

6.3.2. Use Case Configure Ballot

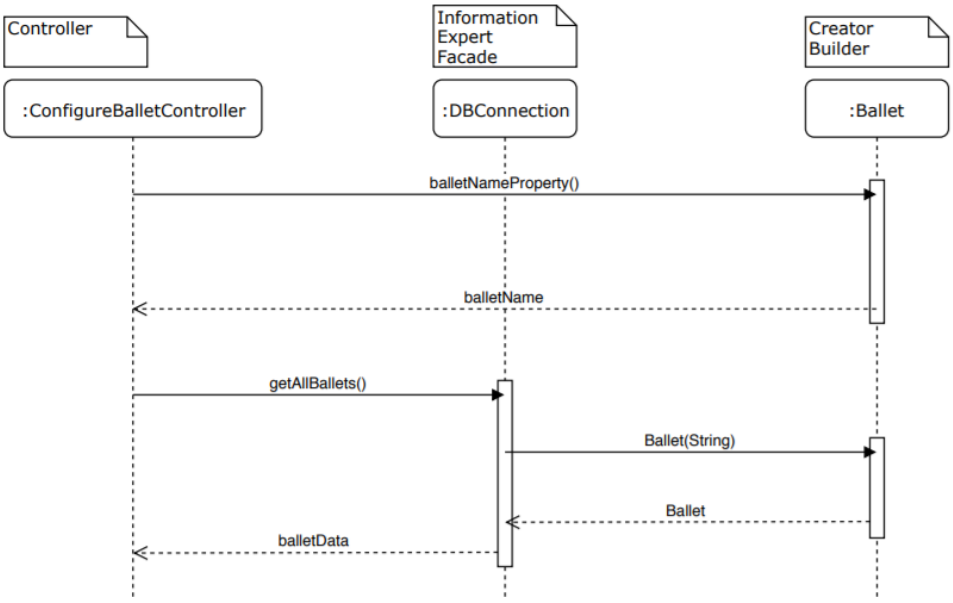

Figure 4.10 Sequence Diagram – Configure Ballot