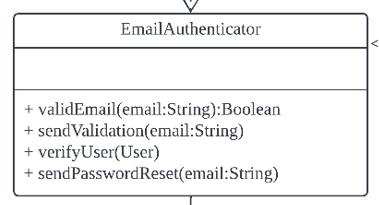
## Black Box – Equivalence Class and Boundary Value Testing

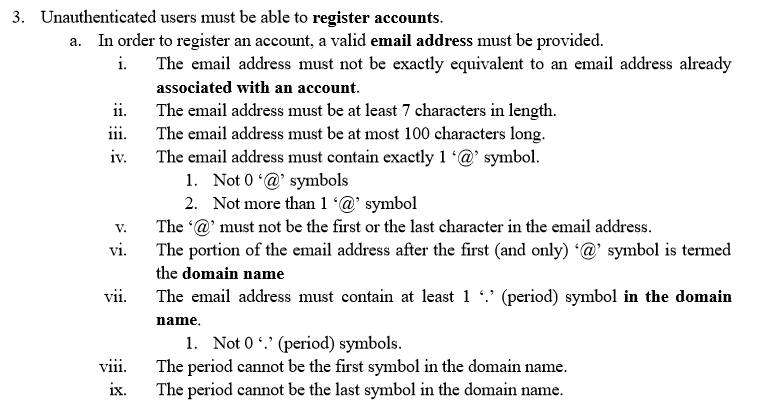
**Design test cases using equivalence class and boundary value testing techniques to test one important control class that implements some important application logic according to the requirements specification.**

What we are testing: The **EmailAuthenticator**.

How do we test it: Manual testing.



Below, we reproduce the relevant functional requirements for validity of an email.





We thus end up with equivalence classes corresponding to these attributes:

1. Length of email address.
2. Number of ‘@’ characters in email address.
3. Is there at least one ‘.’ (period) character after the ‘@’?
4. Is there a ‘.’ (period) character as the first character immediately after the ‘@’?
5. Is there a ‘.’ (period) character as the last character in the email address?

Valid and invalid ECs for these are outlined below.

| Length of email address | <7 | ≥7, ≤100 | >100 |
| --- | --- | --- | --- |
| Equivalence Class | Invalid | Valid | Invalid |

| Number of ‘@’ characters in email address | <1 | =1 | >1 |
| --- | --- | --- | --- |
| Equivalence Class | Invalid | Valid | Invalid |

| Is there at least one ‘.’ (period) character after the ‘@’? | =0 | >0 |
| --- | --- | --- |
| Equivalence Class | Invalid | Valid |

| Is there a ‘.’ (period) character as the first character immediately after the ‘@’? | Yes | No |
| --- | --- | --- |
| Equivalence Class | Invalid | Valid |

| Is there a ‘.’ (period) character as the last character in the email address? | Yes | No |
| --- | --- | --- |
| Equivalence Class | Invalid | Valid |

Let us now design the corresponding test cases, using concepts of EC and BC testing. While our inputs to the algorithm for checking validity of email addresses are multiple, we will only input one string to our app, the email address, thus that is noted in the columns. Our range-based variables are length of email address and number of ‘@’ characters. Our other three variables are discrete. Thus we have two sets of range-based ECs with three ECs each, and three sets of discrete ECs with two ECs each.

## EC-based Testing

| Length of email address | Number of ‘@’ characters in email address | Is there at least one ‘.’ (period) character after the ‘@’? | Is there a ‘.’ (period) character as the first character immediately after the ‘@’? | Is there a ‘.’ (period) character as the last character in the email address? | Email address input | Expected result | Actual result |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 1 | Yes | No | No | ac@b.com | Valid | Valid |
| 5 | 1 | Yes | No | No | a@b.c | Invalid | Invalid |
| 120 | 1 | Yes | No | No | LiMIsPSGeP7NNtLLBzfTx3NSkwgw70tfEmN6FBaFdL.6sGUcyvg  vl5.vYBA5anVYQQ9abGQCF2E6dDNDfEmN6FBaFdL16s@GUcyvg4G6zEf.yJP1jJwCmO.f04 | Invalid | Invalid |
| 9 | 0 | Yes | No | No | fhfhfh.com | Invalid | Invalid |
| 10 | 4 | Yes | No | No | A\_@@@@z.co | Invalid | Invalid |
| 12 | 1 | No | No | No | adhhd@xsfsdf | Invalid | Invalid |
| 15 | 1 | Yes | Yes | No | Ac.s\_Axs@.sfsdf | Invalid | Invalid |
| 17 | 1 | Yes | No | Yes | 9\_4f\df123\_@23sdfs. | Invalid | Invalid |

## BC-based testing

**Length of email address**: Valid boundaries are {7, 100}. Invalid boundaries are {6, 101}

**Number of ‘@’ characters in email address**:

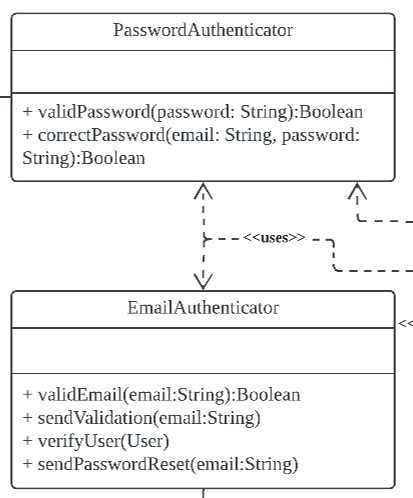
Valid boundary is { 1 }. Invalid boundaries are {0, 2}

The rest are discrete and don’t have boundary values.

| Length of email address | Number of ‘@’ characters in email address | Is there at least one ‘.’ (period) character after the ‘@’? | Is there a ‘.’ (period) character as the first character immediately after the ‘@’? | Is there a ‘.’ (period) character as the last character in the email address? | Email address input | Expected result | Actual result |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 1 | Yes | No | No | a@b.com | Valid | Valid |
| 100 | 1 | Yes | No | No | fkfgkcebezmgtdzvewdqyxbptggzhpkasttmerhvhbuysu@hclyxkzsaffthlymfoiupxrapwzquzrimelhmvhhtsjjnsw.ptwyr | Valid | Valid |
| 6 | 1 | Yes | No | No | ac@b.c | Invalid | Invalid |
| 101 | 1 | Yes | No | No | rdmawzlooxxnxwcornkpznad@kckhtbisrecvzorvekvvxvelvlrmnnndqavkbvnhzgfj.yxhbzviehnpqdf.lux.xzs.obetrcqt | Invalid | Invalid |
| 7 | 0 | Yes | No | No | ffhm.sd | Invalid | Invalid |
| 100 | 2 | Yes | No | No | trmiikcck@berbulrztwhbgad.hbjedtdqbbrgjfugrudzccgb@hmetpswjpvijaxjavuruqauvjpdvbjgpm.wgfo.ylooa.aedd | Invalid | Invalid |
| 7 | 1 | No | No | No | ad@ds1f | Invalid | Invalid |
| 100 | 1 | Yes | Yes | No | kzd11glfzbyyxyaoivxjjrfrmzge.bdgev.oqxrdrzxpktlkr@.gdzsjbwptwxewp33vm.qszdvlhcllqjvk00qj\_\_.jkwe.hzle | Invalid | Invalid |
| 7 | 1 | Yes | No | Yes | 9\_\\@f. | Invalid | Invalid |

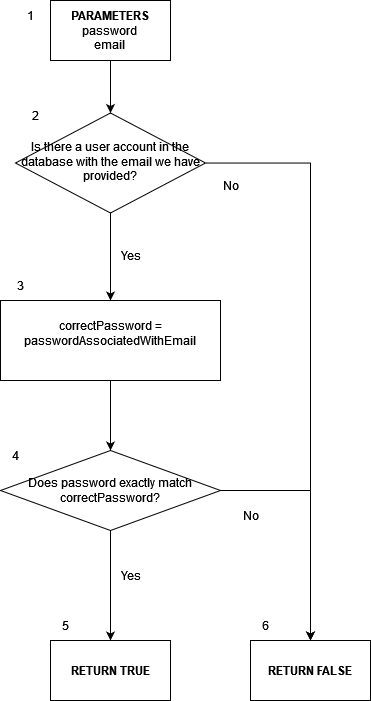
## Basis Path Testing

Again, we use exclusively manual testing.



**correctPassword** and **validEmail** are the two methods we choose to test via basis path testing.

### CFG of correctPassword



**Cyclomatic Complexity: 2 + 1 = 3.**

**Basis path #1 (baseline path):** 1 -> 2 -> 3 -> 4 -> 5.

**Basis path #2:** 1->2->**6**.

**Basis path #2:** 1->2->3->**4->6**.

Before running these test cases, ensure that user accounts with these credentials have been registered and email-verified.

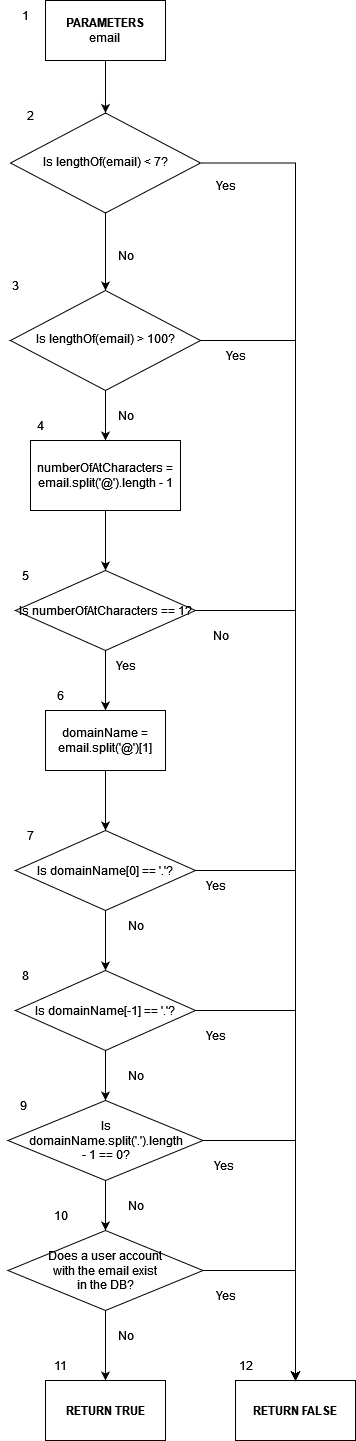
Email: [adakjsakd@example.com](mailto:adakjsakd@gmail.com). Password: 1J@jasdkasfkj

Password: [aosdlasjd@example.com](mailto:aosdlasjd@example.com). Password: M7@skhdaadsa

| Test case #  Aka, basis path # | **Input:** email | **Input:** password | Expected result | Actual result |
| --- | --- | --- | --- | --- |
| 1 | [adakjsakd@example.com](mailto:adakjsakd@gmail.com) | 1J@jasdkasfkj | True | True |
| 2 | [sjd@example.com](mailto:aosdlasjd@example.com) | M7@skhdaadsa | False | False |
| 3 | [aosdlasjd@example.com](mailto:aosdlasjd@example.com) | 9A@AKnkjasa | False | False |

### 

### CFG of validEmail



**Cyclomatic Complexity: 7 + 1 = 8.**

**Basis path #1 (baseline path):** 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 11.

**Basis path #2:** 1-> 2 -> **12**.

**Basis path #3:** 1-> 2 -> 3-> **12**.

**Basis path #4:** 1-> 2 -> 3-> 4 -> 5 -> **12**.

**Basis path #5:** 1-> 2 -> 3-> 4 -> 5 -> 6 -> 7 -> **12**.

**Basis path #6:** 1-> 2 -> 3-> 4 -> 5 -> 6 -> 7 -> 8 -> **12**.

**Basis path #7:** 1-> 2 -> 3-> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> **12**.

**Basis path #7:** 1-> 2 -> 3-> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -**>** 10 **-> 12**.

Before running these test cases, ensure that user accounts with these credentials have been registered and email-verified.

Email: [adakjsakd@example.com](mailto:adakjsakd@gmail.com). Password: 1J@jasdkasfkj

Password: [aosdlasjd@example.com](mailto:aosdlasjd@example.com). Password: M7@skhdaadsa

| Test case #  Aka, basis path # | **Input:** email | Expected result | Actual result |
| --- | --- | --- | --- |
| 1 | [sjd@example.com](mailto:aosdlasjd@example.com) | True | True |
| 2 | abc | False | False |
| 3 | abcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabcabc | False | False |
| 4 | abc@dasaa@sdg | False | False |
| 5 | ashdjsahdj@.aass | False | False |
| 6 | asdkasdkk@aass. | False | False |
| 7 | ajhsdj23hi@djsk | False | False |
| 8 | [adakjsakd@example.com](mailto:adakjsakd@gmail.com) | False | False |