In [1]:
```
import numpy as np  #linear algebra
import pandas as pd # a data processing and CSV I/O library
# Data Visualization
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style='white', color_codes=True)
```

In [2]:
```
data=pd.read_csv("Job_Placement_Dataset.CSV")
```

In [3]:
```
data
```

Out[3]:

	gender	ssc_percentage	ssc_board	hsc_percentage	hsc_board	hsc_subject	degree_percentage	undergrad_degree	work_experience	e
0	M	67.00	Others	91.00	Others	Commerce	58.00	Sci&Tech	0	
1	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech	1	
2	M	65.00	Central	68.00	Central	Arts	64.00	Comm&Mgmt	0	
3	M	56.00	Central	52.00	Central	Science	52.00	Sci&Tech	0	
4	M	85.80	Central	73.60	Central	Commerce	73.30	Comm&Mgmt	0	
...	...	...	...	...	...	...	...	...	...	
210	M	80.60	Others	82.00	Others	Commerce	77.60	Comm&Mgmt	0	
211	M	58.00	Others	60.00	Others	Science	72.00	Sci&Tech	0	
212	M	67.00	Others	67.00	Others	Commerce	73.00	Comm&Mgmt	1	
213	F	74.00	Others	66.00	Others	Commerce	58.00	Comm&Mgmt	0	
214	M	62.00	Central	58.00	Others	Science	53.00	Comm&Mgmt	0	

215 rows × 13 columns

In [4]:
```
data.head()
```

Out[4]:

	gender	ssc_percentage	ssc_board	hsc_percentage	hsc_board	hsc_subject	degree_percentage	undergrad_degree	work_experience	em
0	M	67.00	Others	91.00	Others	Commerce	58.00	Sci&Tech	0	
1	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech	1	
2	M	65.00	Central	68.00	Central	Arts	64.00	Comm&Mgmt	0	
3	M	56.00	Central	52.00	Central	Science	52.00	Sci&Tech	0	
4	M	85.80	Central	73.60	Central	Commerce	73.30	Comm&Mgmt	0	

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 215 entries, 0 to 214
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   gender                215 non-null    object 
 1   ssc_percentage        215 non-null    float64
 2   ssc_board             215 non-null    object 
 3   hsc_percentage        215 non-null    float64
 4   hsc_board             215 non-null    object 
 5   hsc_subject           215 non-null    object 
 6   degree_percentage     215 non-null    float64
 7   undergrad_degree      215 non-null    object 
 8   work_experience       215 non-null    int64  
 9   emp_test_percentage   215 non-null    float64
 10  specialisation        215 non-null    object 
 11  msc_percent           215 non-null    float64
 12  status                215 non-null    object 
dtypes: float64(5), int64(1), object(7)
memory usage: 22.0+ KB
```

In [6]:
```
data.info
```

Out[6]:

```
<bound method DataFrame.info of     gender  ssc_percentage ssc_board  hsc_percentage hsc_board hsc_subject  \
0        M           67.00    Others           91.00    Others    Commerce
1        M           79.33   Central           78.33    Others     Science
2        M           65.00   Central           68.00   Central        Arts
3        M           56.00   Central           52.00   Central     Science
4        M           85.80   Central           73.60   Central    Commerce
..     ...             ...       ...             ...       ...         ...
210      M           80.60    Others           82.00    Others    Commerce
211      M           58.00    Others           60.00    Others     Science
212      M           67.00    Others           67.00    Others    Commerce
213      F           74.00    Others           66.00    Others    Commerce
214      M           62.00   Central           58.00    Others     Science

     degree_percentage undergrad_degree  work_experience  emp_test_percentage  \
0                58.00         Sci&Tech                0                 55.0
1                77.48         Sci&Tech                1                 86.5
2                64.00        Comm&Mgmt                0                 75.0
3                52.00         Sci&Tech                0                 66.0
4                73.30        Comm&Mgmt                0                 96.8
..                 ...              ...              ...                  ...
210              77.60        Comm&Mgmt                0                 91.0
211              72.00         Sci&Tech                0                 74.0
212              73.00        Comm&Mgmt                1                 59.0
213              58.00        Comm&Mgmt                0                 70.0
214              53.00        Comm&Mgmt                0                 89.0

      specialisation  msc_percent      status
0         Statistics        58.80      Placed
1    Computerscience        66.28      Placed
2    Computerscience        57.80      Placed
3         Statistics        59.43  Not Placed
4    Computerscience        55.50      Placed
..               ...          ...         ...
210  Computerscience        74.49      Placed
211  Computerscience        53.62      Placed
212  Computerscience        69.72      Placed
213       Statistics        60.23      Placed
214       Statistics        60.22  Not Placed

[215 rows x 13 columns]>
```

In [7]:

```
data.describe()
```

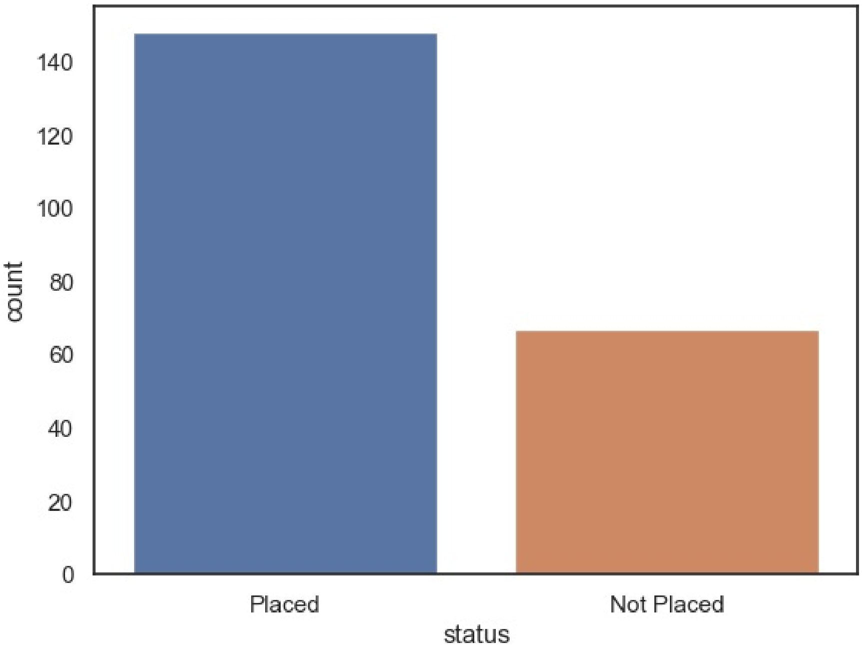
Out[7]:

	ssc_percentage	hsc_percentage	degree_percentage	work_experience	emp_test_percentage	msc_percent
count	215.000000	215.000000	215.000000	215.000000	215.000000	215.000000
mean	67.303395	66.333163	66.370186	0.344186	72.100558	62.278186
std	10.827205	10.897509	7.358743	0.476211	13.275956	5.833385
min	40.890000	37.000000	50.000000	0.000000	50.000000	51.210000
25%	60.600000	60.900000	61.000000	0.000000	60.000000	57.945000
50%	67.000000	65.000000	66.000000	0.000000	71.000000	62.000000
75%	75.700000	73.000000	72.000000	1.000000	83.500000	66.255000
max	89.400000	97.700000	91.000000	1.000000	98.000000	77.890000

In [8]:

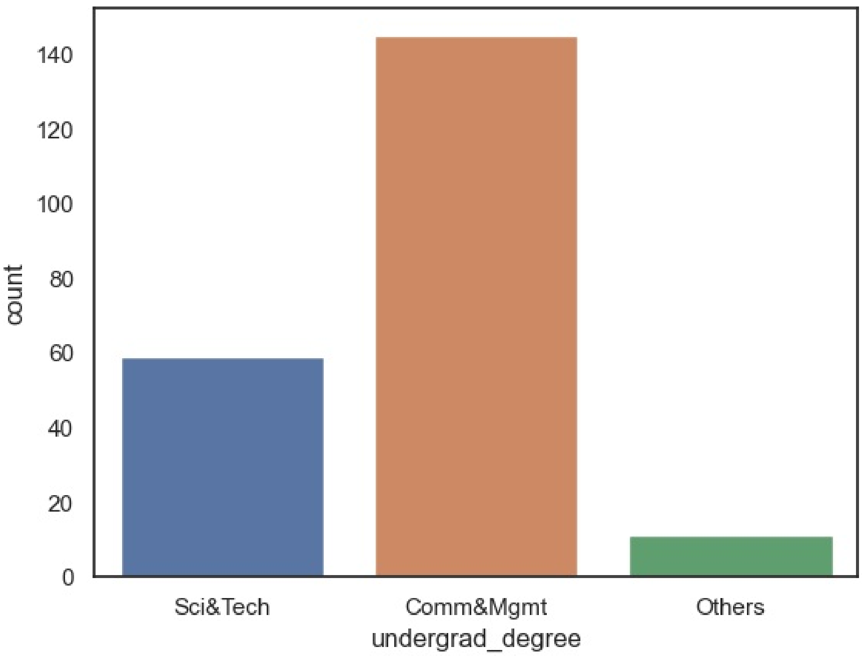
```
print(data['status'].value_counts())
_ = sns.countplot(x='status', data=data)
```

```
Placed        148
Not Placed     67
Name: status, dtype: int64
```



In [9]:
```
print(data['undergrad_degree'].value_counts())
_ = sns.countplot(x='undergrad_degree', data=data)
```

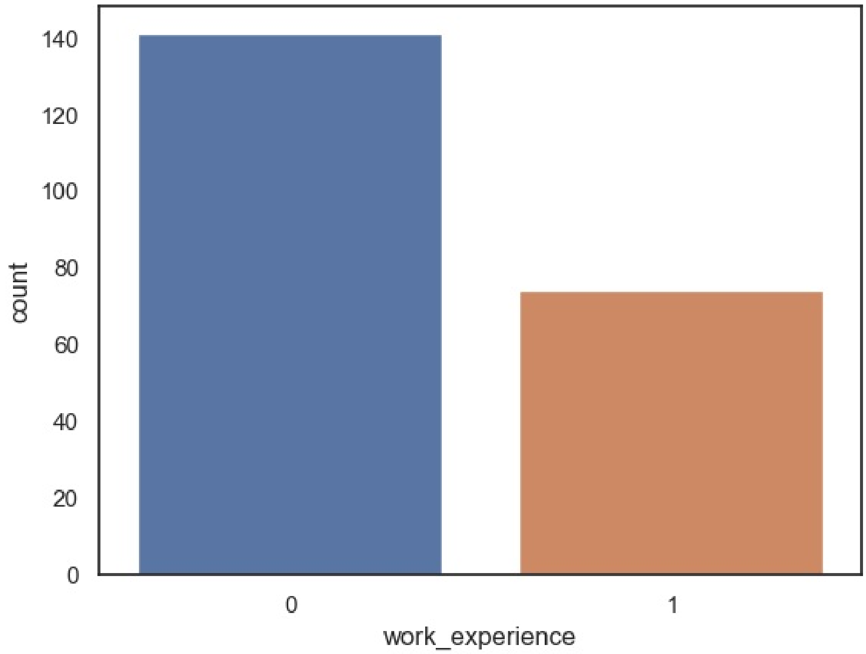
```
Comm&Mgmt    145
Sci&Tech      59
Others        11
Name: undergrad_degree, dtype: int64
```



In [10]:
```
print(data['work_experience'].value_counts())
```

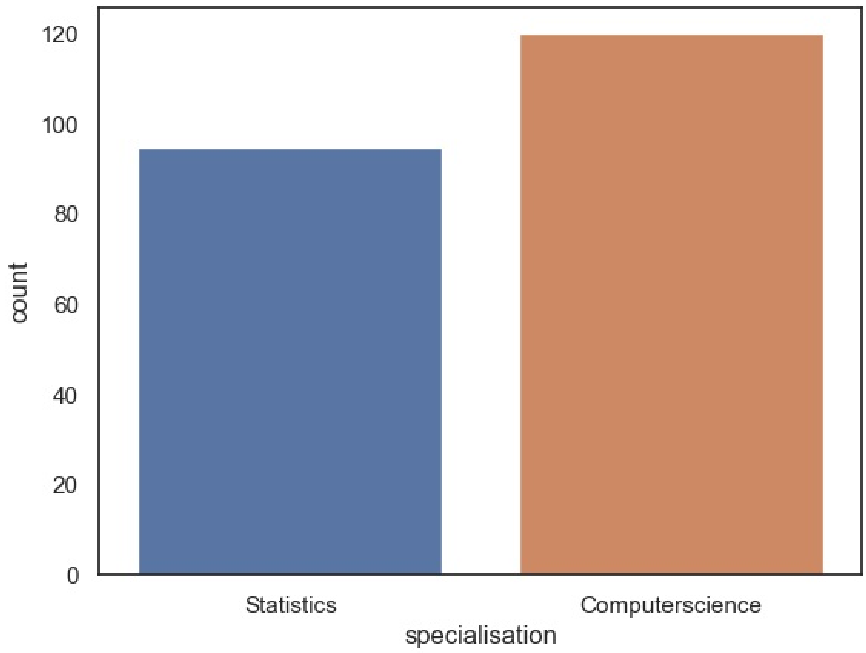
```
_ = sns.countplot(x='work_experience', data=data)
```

```
0    141
1     74
Name: work_experience, dtype: int64
```



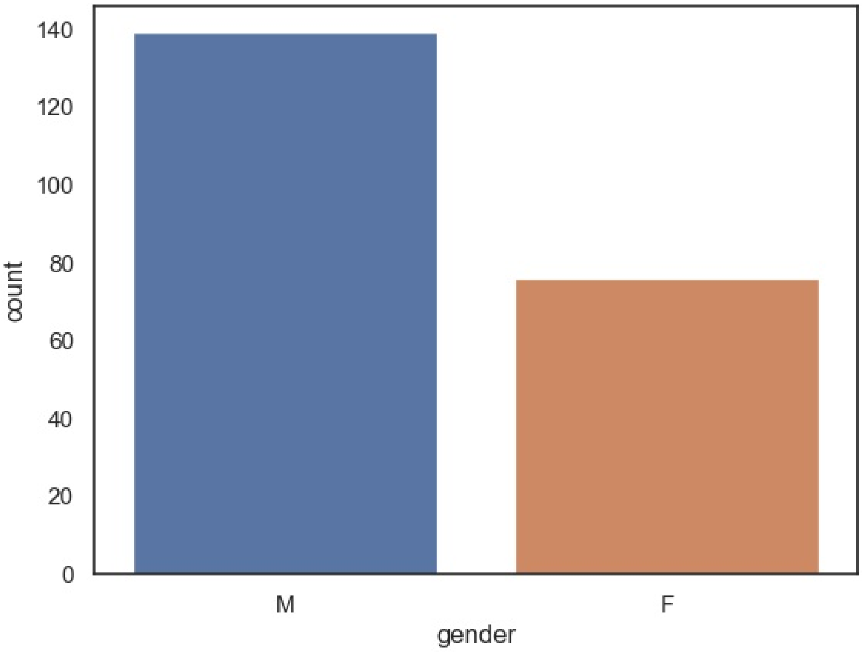
In [11]:
```
print(data['specialisation'].value_counts())
_ = sns.countplot(x='specialisation', data=data)
```

```
Computerscience    120
Statistics          95
Name: specialisation, dtype: int64
```



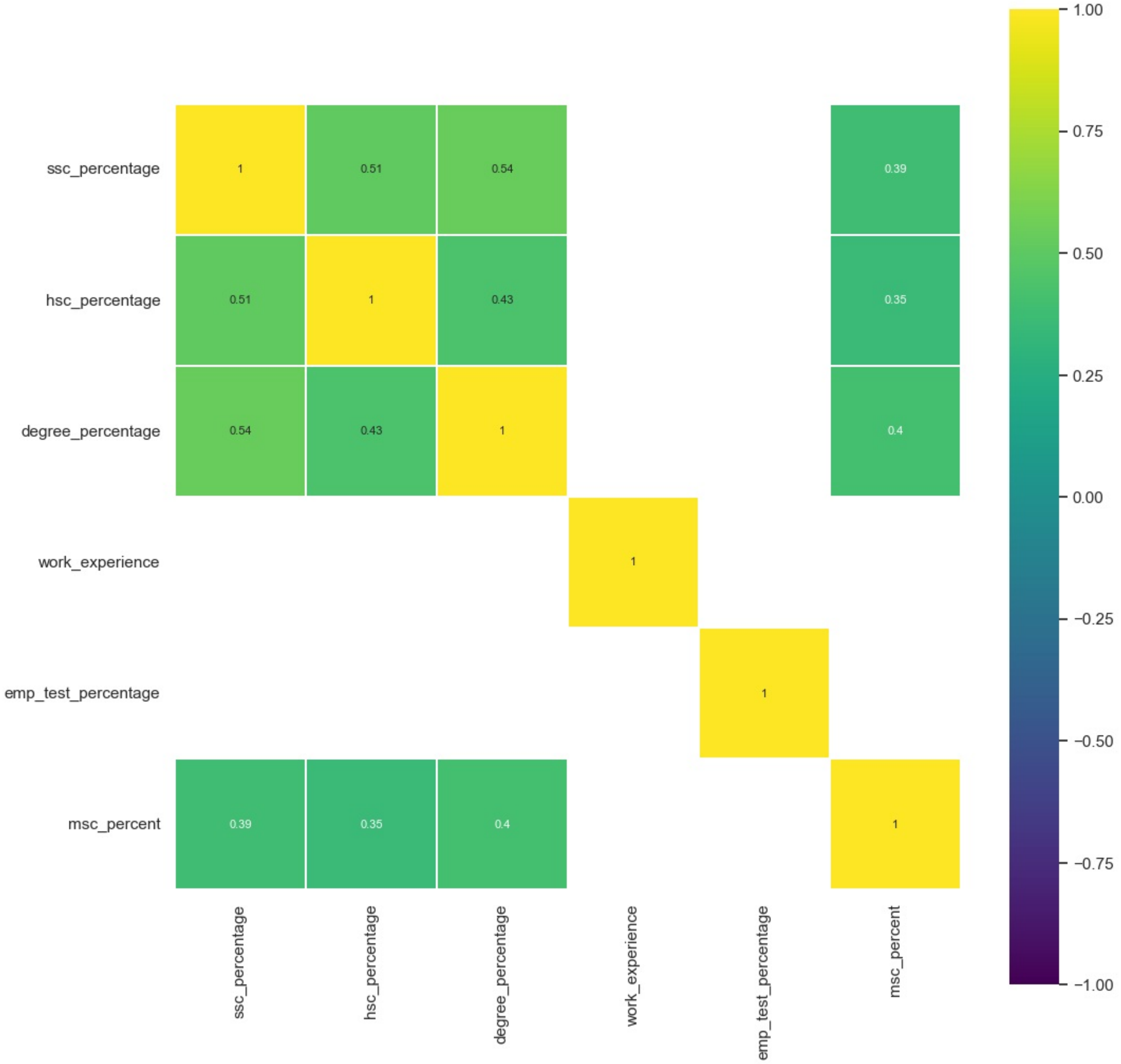
In [12]:
```
print(data['gender'].value_counts())
_ = sns.countplot(x='gender', data=data)
```

```
M    139
F     76
Name: gender, dtype: int64
```



In [13]:

```
corr = data.corr()
plt.figure(figsize=(12, 12))

sns.heatmap(corr[(corr >= 0.3) | (corr <= -0.3)], cmap='viridis', vmax=1.0, vmin=-1.0, linewidths=0.1,annot=Tru
```

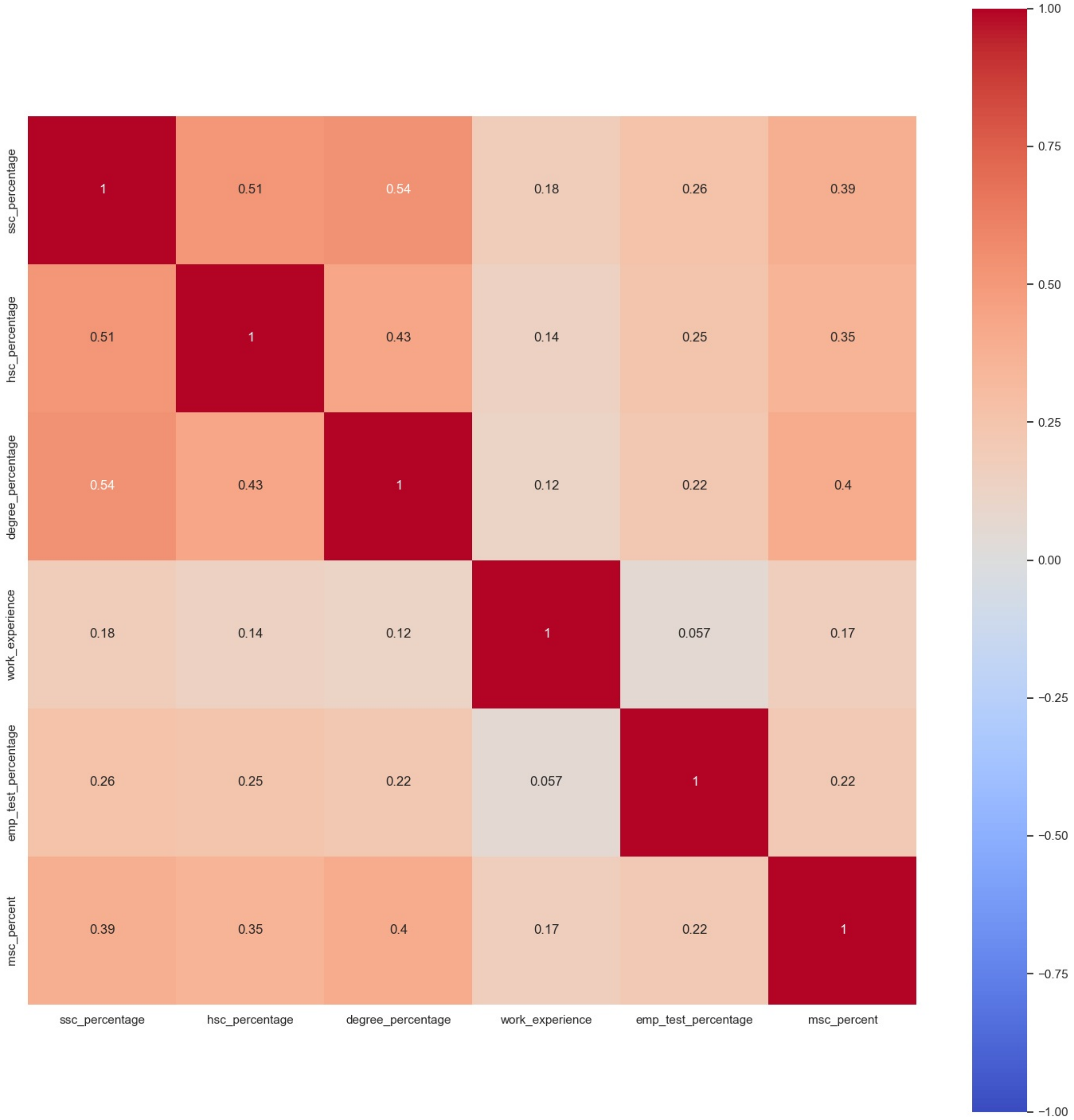


In [14]:

```
plt.figure(figsize=(18,18))
sns.heatmap(data.corr("pearson"),vmin=-1, vmax=1,cmap='coolwarm',annot=True, square=True)
```

Out[14]:

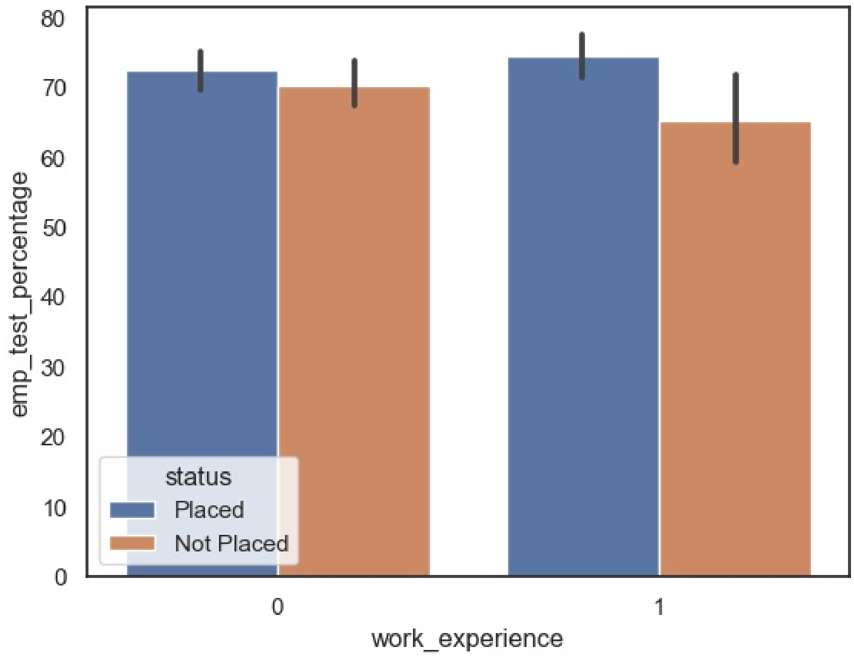
```
<AxesSubplot:>
```



ssc_percentage
hsc_percentage
degree_percentage
work_experience
emp_test_percentage
msc_percent
1 0.51 0.54 0.18 0.26 0.39
0.51 1 0.43 0.14 0.25 0.35
0.54 0.43 1 0.12 0.22 0.4
0.18 0.14 0.12 1 0.057 0.17
0.26 0.25 0.22 0.057 1 0.22
0.39 0.35 0.4 0.17 0.22 1
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

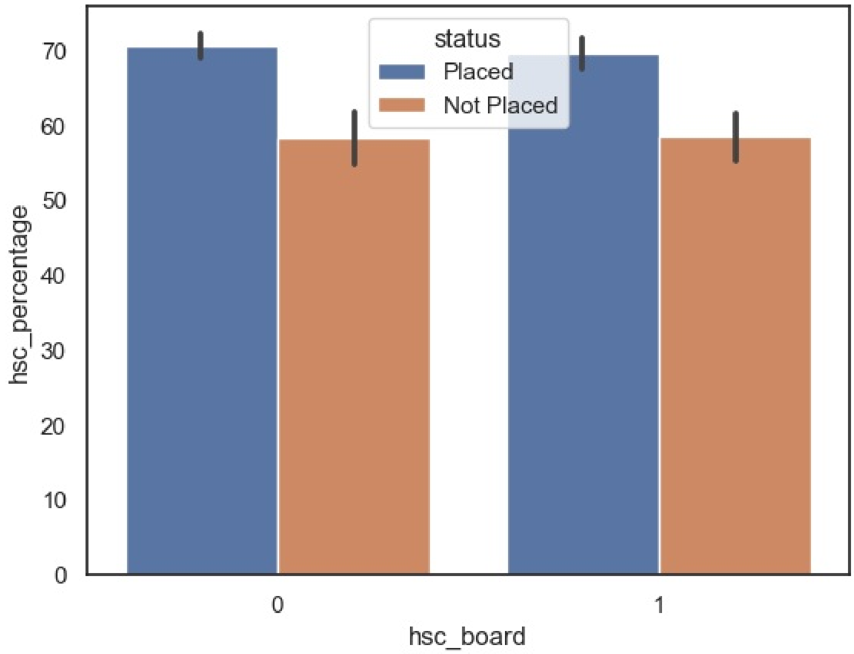
In [19]:
```
#Bar plot based on the 'work_experience' and 'emp_test_percentage' on the basis of 'status'
sns.barplot(x='work_experience',y='emp_test_percentage',hue='status',data=data)
```

Out[19]: <AxesSubplot:xlabel='work_experience', ylabel='emp_test_percentage'>



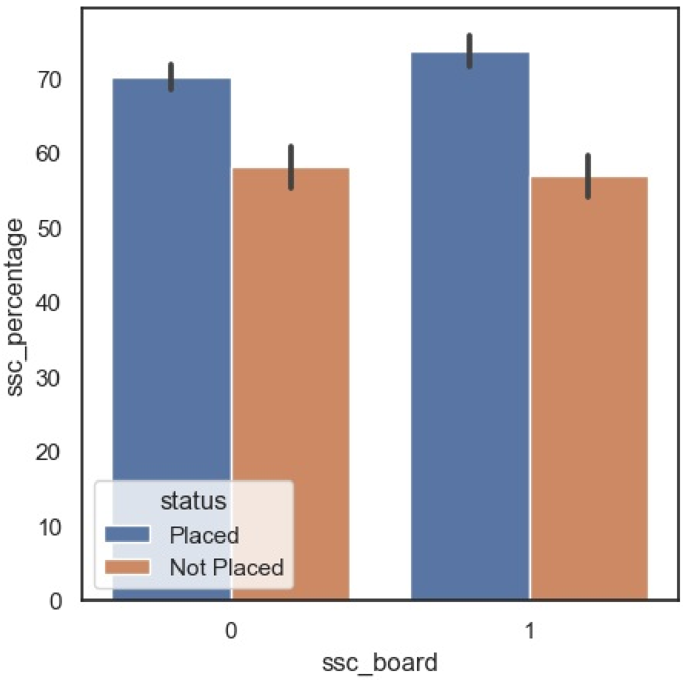
In [20]:
```
#Bar plot based on the 'hsc_board' and 'hsc_percentage' on the basis of 'status'
sns.barplot(x='hsc_board',y='hsc_percentage',hue='status',data=data)
```

Out[20]: <AxesSubplot:xlabel='hsc_board', ylabel='hsc_percentage'>



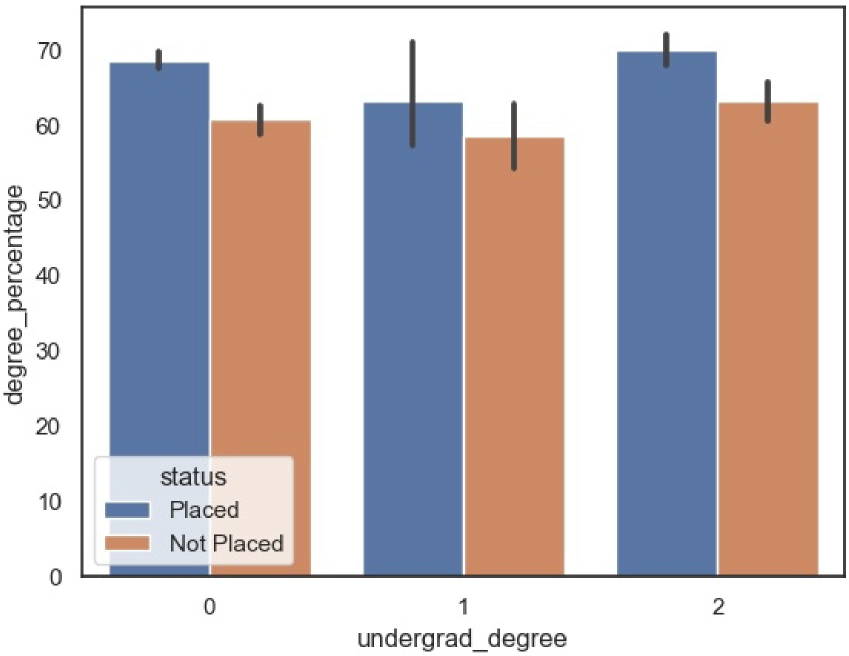
In [21]:
```
#Bar plot based on the 'ssc_board' and 'ssc_percentage' on the basis of 'status'
plt.figure(figsize=(5,5))
sns.barplot(x='ssc_board',y='ssc_percentage',hue='status',data=data)
```

Out[21]: <AxesSubplot:xlabel='ssc_board', ylabel='ssc_percentage'>



In [24]:
```
#Bar plot based on the 'undergrad_degree' and 'degree_percentagee' on the basis of 'status'
sns.barplot(x='undergrad_degree',y='degree_percentage',hue='status',data=data)
```

Out[24]:
```
<AxesSubplot:xlabel='undergrad_degree', ylabel='degree_percentage'>
```

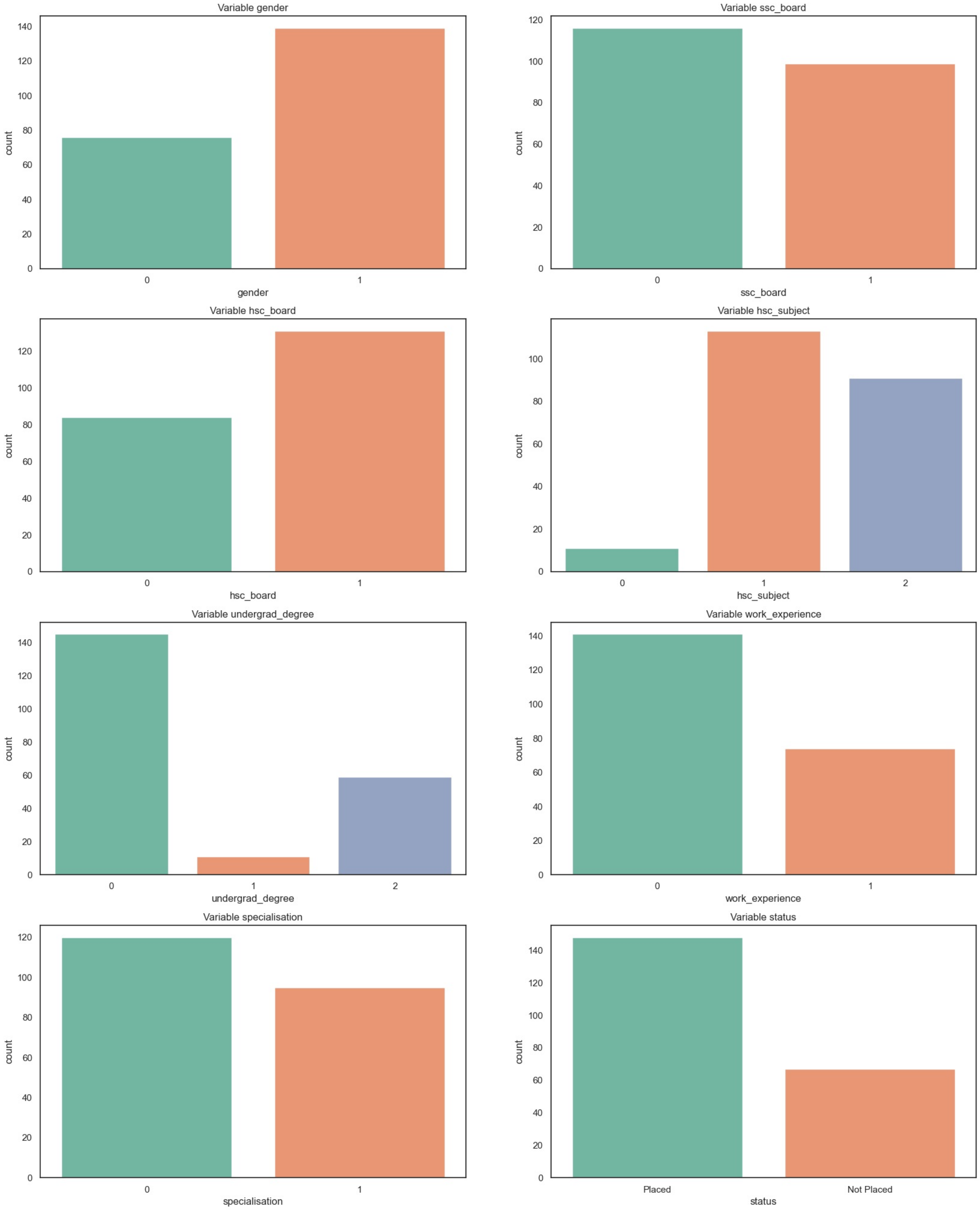


In [26]:
```
plt.figure(figsize = (20,25))

plt.subplot(4,2,1)
plt.gca().set_title('Variable gender')
sns.countplot(x = 'gender', palette = 'Set2', data = data)

plt.subplot(4,2,2)
plt.gca().set_title('Variable ssc_board')
sns.countplot(x = 'ssc_board', palette = 'Set2', data = data)
```

```python
plt.subplot(4,2,3)
plt.gca().set_title('Variable hsc_board')
sns.countplot(x = 'hsc_board', palette = 'Set2', data = data)

plt.subplot(4,2,4)
plt.gca().set_title('Variable hsc_subject')
sns.countplot(x = 'hsc_subject', palette = 'Set2', data = data)

plt.subplot(4,2,5)
plt.gca().set_title('Variable undergrad_degree')
sns.countplot(x = 'undergrad_degree', palette = 'Set2', data = data)

plt.subplot(4,2,6)
plt.gca().set_title('Variable work_experience')
sns.countplot(x = 'work_experience', palette = 'Set2', data = data)

plt.subplot(4,2,7)
plt.gca().set_title('Variable specialisation')
sns.countplot(x = 'specialisation', palette = 'Set2', data = data)

plt.subplot(4,2,8)
plt.gca().set_title('Variable status')
sns.countplot(x = 'status', palette = 'Set2', data = data)
```

Out[26]:

```
<AxesSubplot:title={'center':'Variable status'}, xlabel='status', ylabel='count'>
```

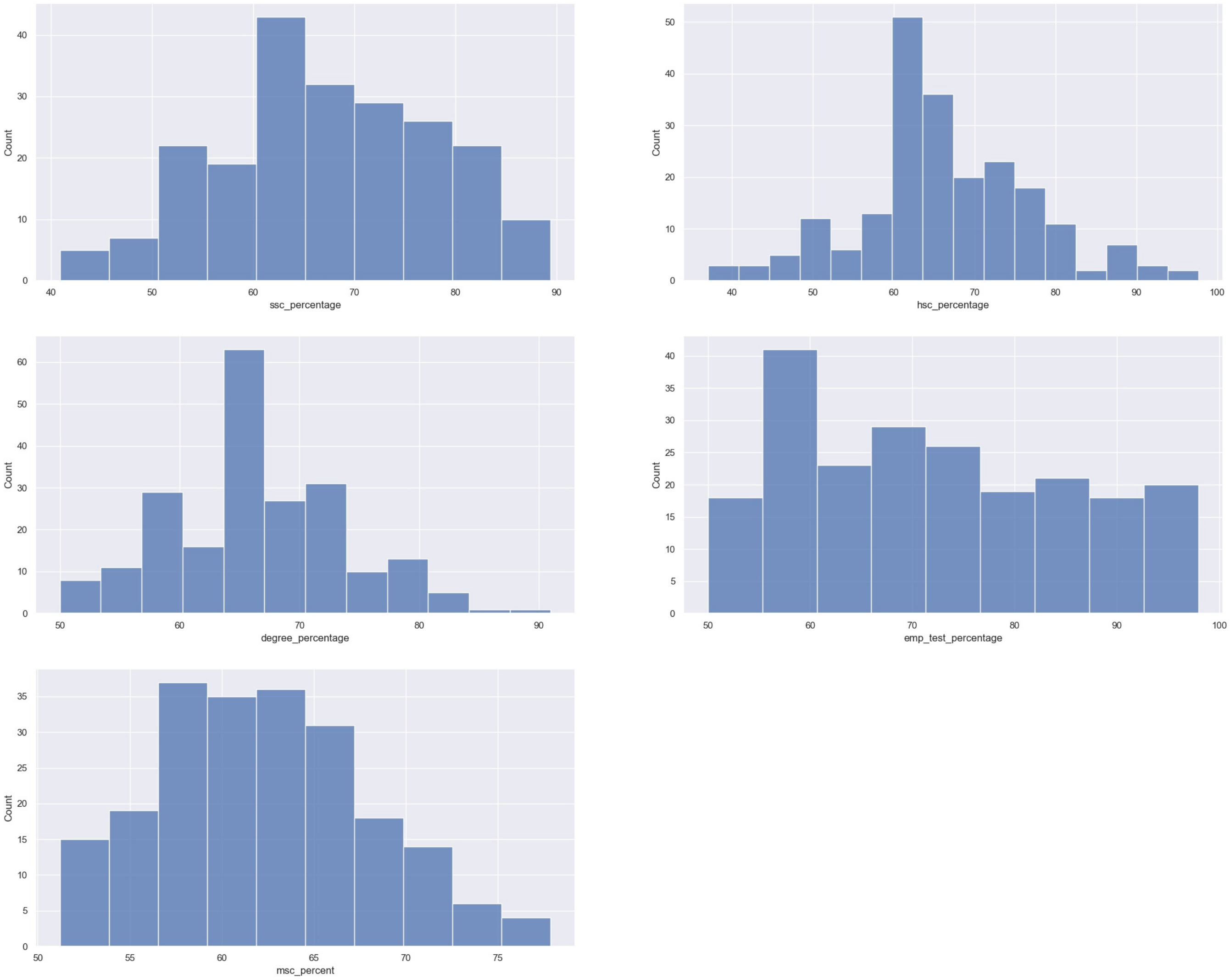


In [27]:

```
plt.figure(figsize = (25,20))
sns.set(color_codes = True)

plt.subplot(3,2,1)
sns.histplot(data['ssc_percentage'], kde = False)

plt.subplot(3,2,2)
sns.histplot(data['hsc_percentage'], kde = False)

plt.subplot(3,2,3)
sns.histplot(data['degree_percentage'], kde = False)

plt.subplot(3,2,4)
sns.histplot(data['emp_test_percentage'], kde = False)

plt.subplot(3,2,5)
sns.histplot(data['msc_percent'], kde = False)
```

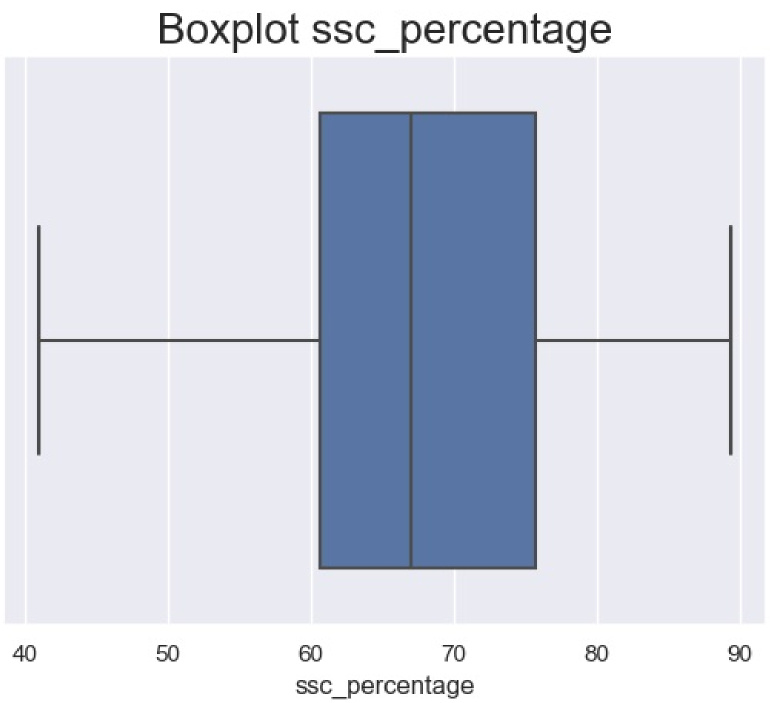
Out[27]:

```
<AxesSubplot:xlabel='msc_percent', ylabel='Count'>
```



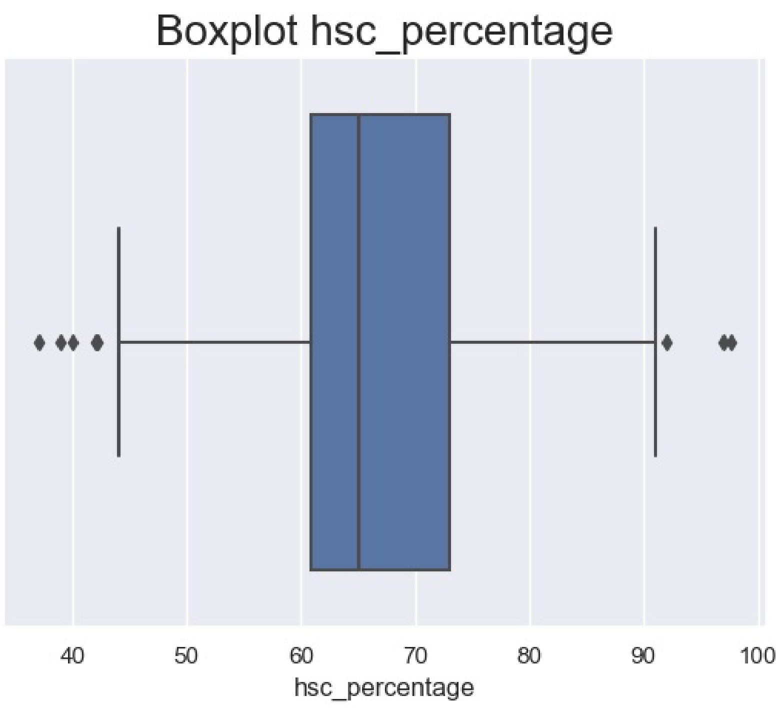
In [28]:
```
plt.title("Boxplot ssc_percentage", fontdict = {'fontsize': 20})
sns.boxplot(x=data["ssc_percentage"])
```

Out[28]: <AxesSubplot:title={'center':'Boxplot ssc_percentage'}, xlabel='ssc_percentage'>



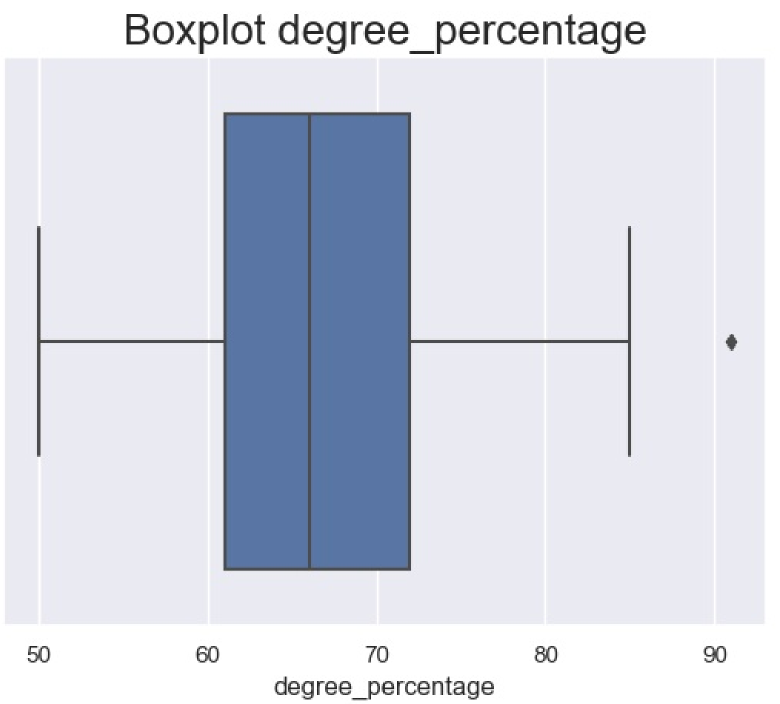
In [29]:
```
plt.title("Boxplot hsc_percentage", fontdict = {'fontsize': 20})
sns.boxplot(x=data
            ["hsc_percentage"])
```

Out[29]: <AxesSubplot:title={'center':'Boxplot hsc_percentage'}, xlabel='hsc_percentage'>



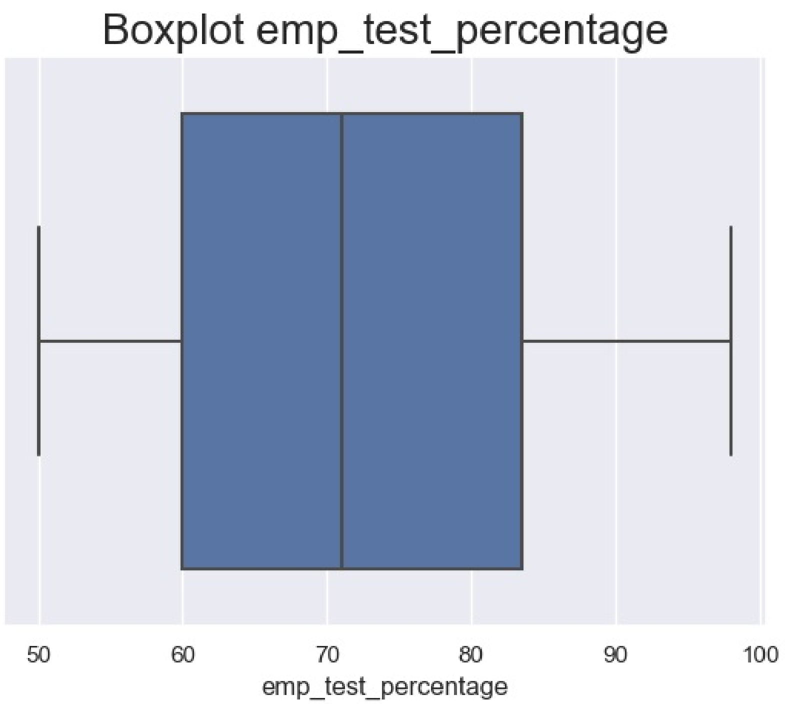
In [30]:
```
plt.title("Boxplot degree_percentage", fontdict = {'fontsize': 20})
sns.boxplot(x=data["degree_percentage"])
```

Out[30]:
```
<AxesSubplot:title={'center':'Boxplot degree_percentage'}, xlabel='degree_percentage'>
```



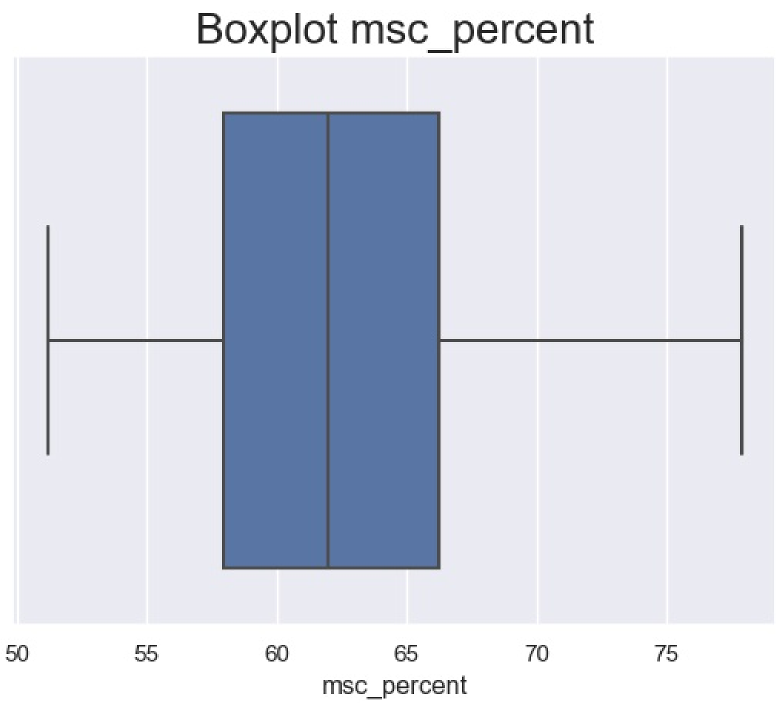
In [31]:
```
plt.title("Boxplot emp_test_percentage", fontdict = {'fontsize': 20})
sns.boxplot(x=data["emp_test_percentage"])
```

Out[31]:
```
<AxesSubplot:title={'center':'Boxplot emp_test_percentage'}, xlabel='emp_test_percentage'>
```



In [32]:
```
plt.title("Boxplot msc_percent", fontdict = {'fontsize': 20})
sns.boxplot(x=data["msc_percent"])
```

Out[32]:
```
<AxesSubplot:title={'center':'Boxplot msc_percent'}, xlabel='msc_percent'>
```

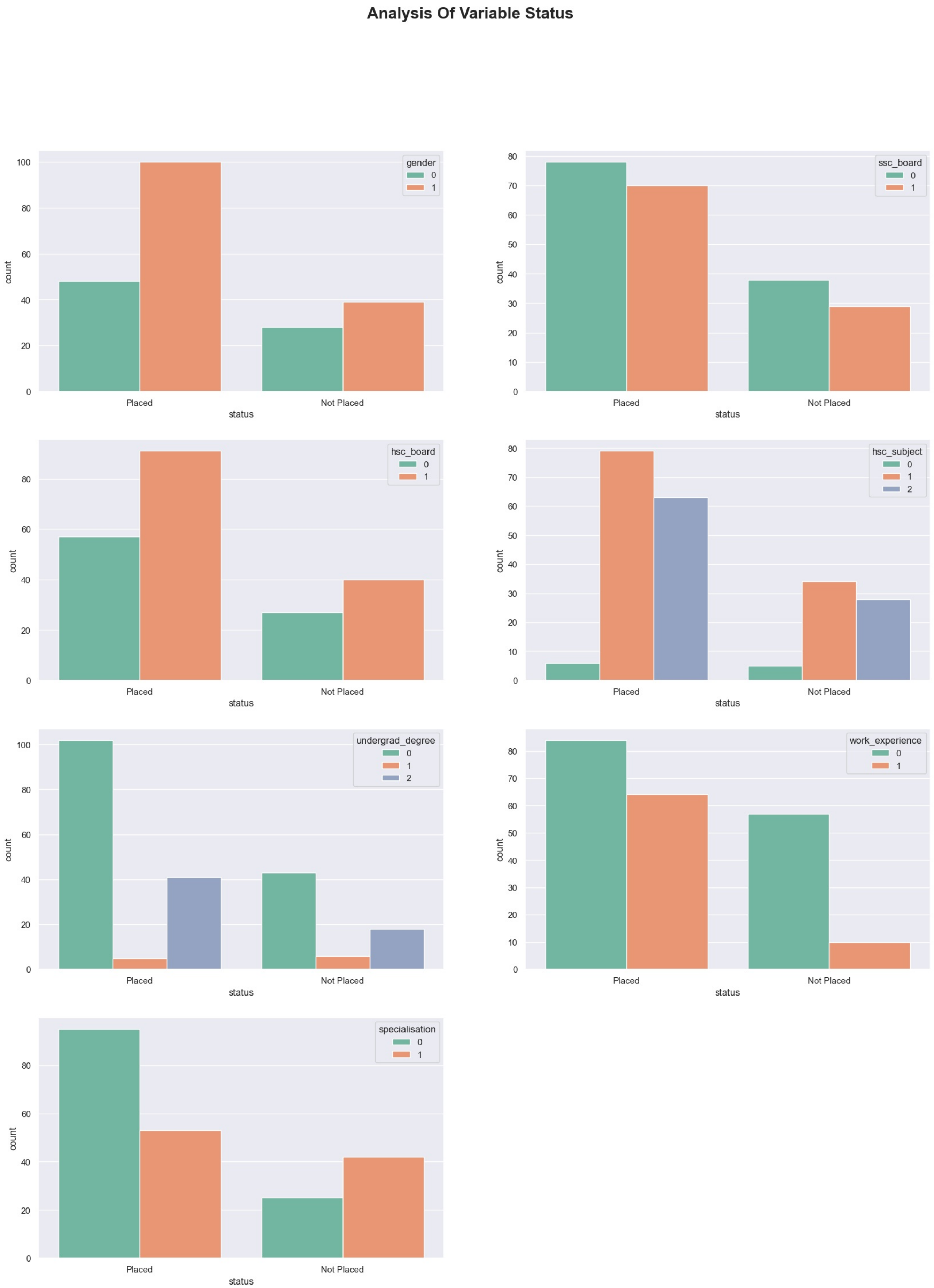


In [33]:
```
plt.figure(figsize = (20, 25))
plt.suptitle("Analysis Of Variable Status",fontweight="bold", fontsize=20)

plt.subplot(4,2,1)
sns.countplot(x = 'status', hue = 'gender', palette = 'Set2', data = data)

plt.subplot(4,2,2)
```

```python
sns.countplot(x = 'status', hue = 'ssc_board', palette = 'Set2', data = data)

plt.subplot(4,2,3)
sns.countplot(x = 'status', hue = 'hsc_board', palette = 'Set2', data = data)

plt.subplot(4,2,4)
sns.countplot(x = 'status', hue = 'hsc_subject', palette = 'Set2', data = data)

plt.subplot(4,2,5)
sns.countplot(x = 'status', hue = 'undergrad_degree', palette = 'Set2', data = data)

plt.subplot(4,2,6)
sns.countplot(x = 'status', hue = 'work_experience', palette = 'Set2', data = data)

plt.subplot(4,2,7)
sns.countplot(x = 'status', hue = 'specialisation', palette = 'Set2', data = data)
```

Out[33]:

```
<AxesSubplot:xlabel='status', ylabel='count'>
```



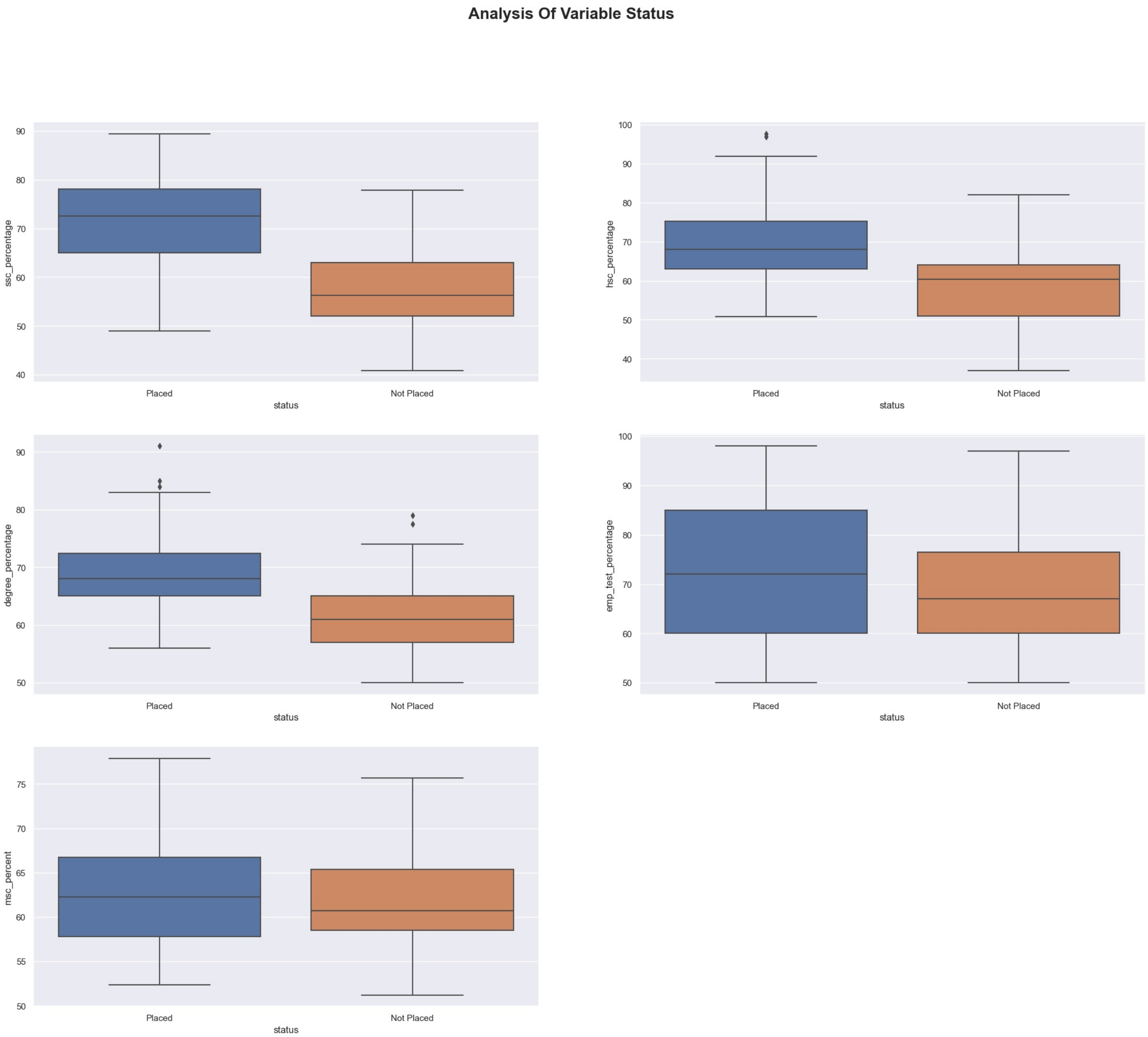
In [35]:

```
plt.figure(figsize = (25, 20))
plt.suptitle("Analysis Of Variable Status",fontweight="bold", fontsize=20)

plt.subplot(3,2,1)
sns.boxplot(x="status", y="ssc_percentage", data=data)

plt.subplot(3,2,2)
sns.boxplot(x="status", y="hsc_percentage", data=data)

plt.subplot(3,2,3)
sns.boxplot(x="status", y="degree_percentage", data=data)
```

```python
plt.subplot(3,2,4)
sns.boxplot(x="status", y="emp_test_percentage", data=data)

plt.subplot(3,2,5)
sns.boxplot(x="status", y="msc_percent", data=data)
```

Out[35]:

```
<AxesSubplot:xlabel='status', ylabel='msc_percent'>
```

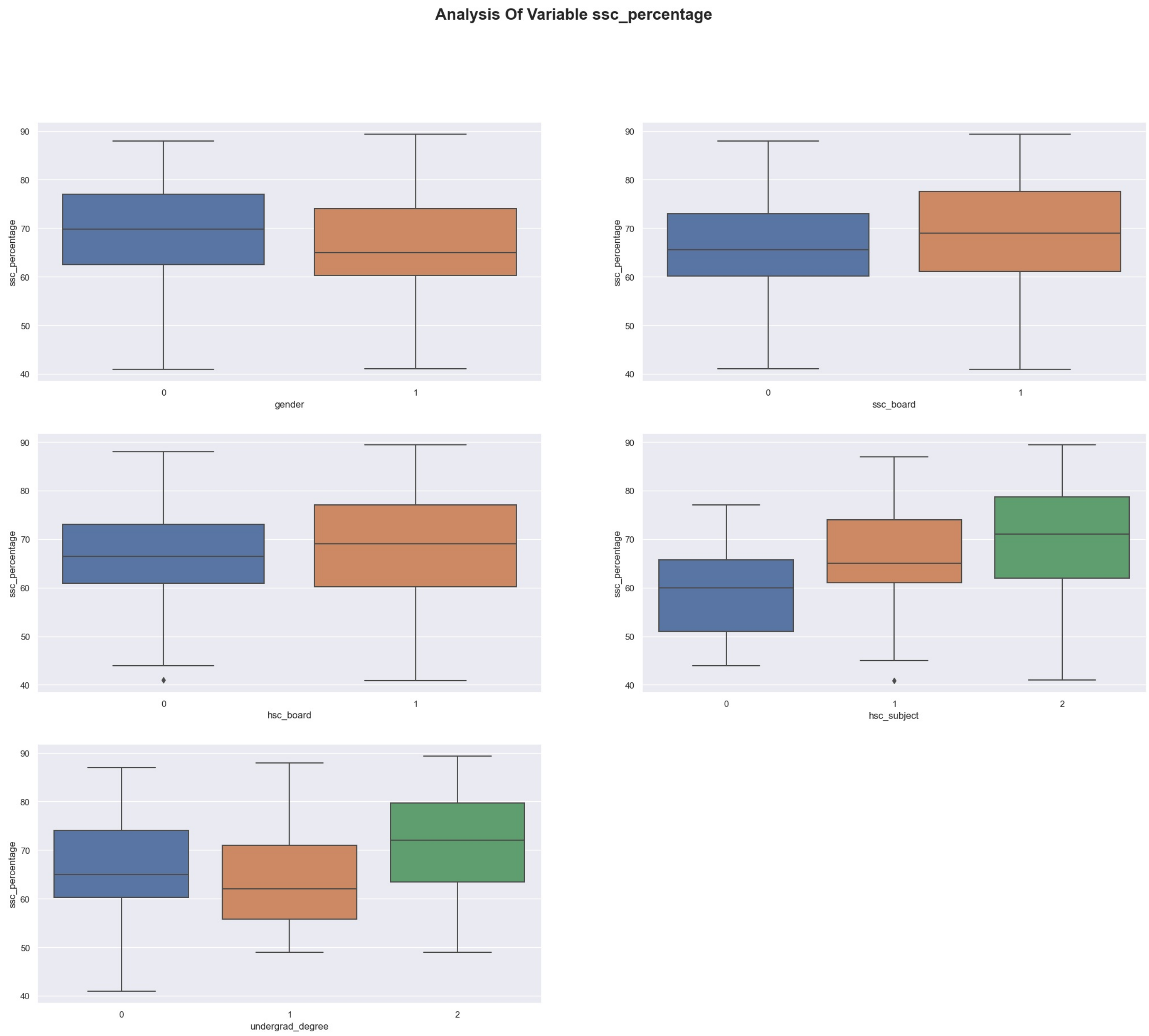


In [36]:

```python
plt.figure(figsize = (25, 20))
plt.suptitle("Analysis Of Variable ssc_percentage",fontweight="bold", fontsize=20)

plt.subplot(3,2,1)
sns.boxplot(x="gender", y="ssc_percentage", data=data)

plt.subplot(3,2,2)
sns.boxplot(x="ssc_board", y="ssc_percentage", data=data)

plt.subplot(3,2,3)
sns.boxplot(x="hsc_board", y="ssc_percentage", data=data)

plt.subplot(3,2,4)
sns.boxplot(x="hsc_subject", y="ssc_percentage", data=data)

plt.subplot(3,2,5)
sns.boxplot(x="undergrad_degree", y="ssc_percentage", data=data)
```

Out[36]:

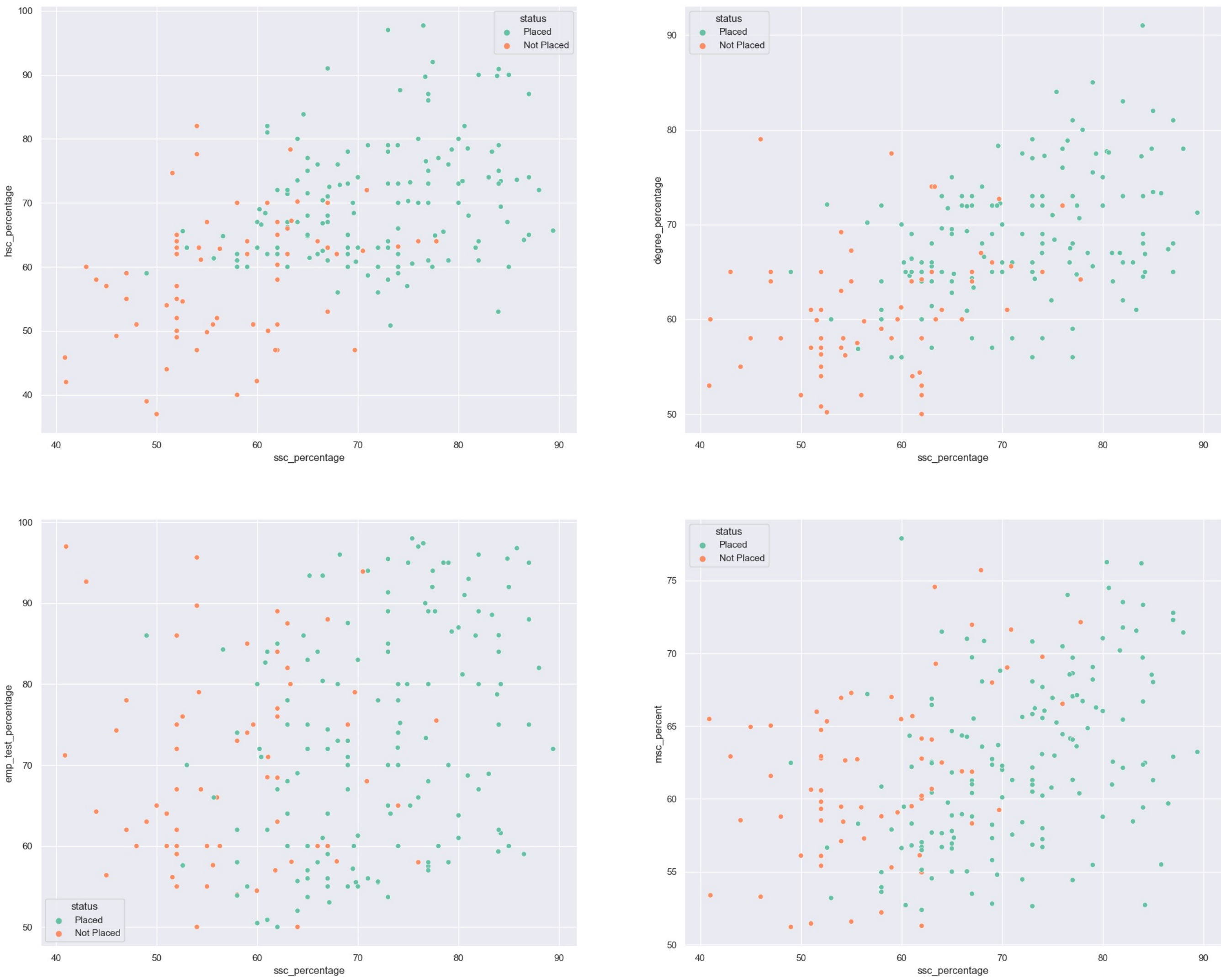
```
<AxesSubplot:xlabel='undergrad_degree', ylabel='ssc_percentage'>
```



In [38]:
```
plt.figure(figsize = (25, 20))
plt.suptitle("Analysis Of Variable ssc_percentage",fontweight="bold", fontsize=20)

plt.subplot(2,2,1)
sns.scatterplot(data=data, x="ssc_percentage", y="hsc_percentage", palette = 'Set2', hue = 'status')

plt.subplot(2,2,2)
sns.scatterplot(data=data, x="ssc_percentage", y="degree_percentage", palette = 'Set2', hue = 'status')

plt.subplot(2,2,3)
sns.scatterplot(data=data, x="ssc_percentage", y="emp_test_percentage", palette = 'Set2', hue = 'status')

plt.subplot(2,2,4)
sns.scatterplot(data=data, x="ssc_percentage", y="msc_percent", palette = 'Set2', hue = 'status')
```

Out[38]:
```
<AxesSubplot:xlabel='ssc_percentage', ylabel='msc_percent'>
```

Analysis Of Variable ssc_percentage



In [40]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data['specialisation']=le.fit_transform(data['specialisation'])
data['undergrad_degree']=le.fit_transform(data['undergrad_degree'])
data['hsc_subject']=le.fit_transform(data['hsc_subject'])
data['hsc_board']=le.fit_transform(data['hsc_board'])
data['ssc_board']=le.fit_transform(data['ssc_board'])
data['gender']=le.fit_transform(data['gender'])
data['work_experience']=le.fit_transform(data['work_experience'])
```

In [41]:
```
X = data.drop('status',axis=1)
y = data[['status']]
```

In [42]:
```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn import svm
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score , classification_report, ConfusionMatrixDisplay,precision_score,reca

X_train , X_test , y_train , y_test = train_test_split(X , y , test_size=0.2,random_state=42)
```

In [43]:
```
models={
    "Logisitic Regression" :LogisticRegression(max_iter=20000),
    "Decision Tree" :DecisionTreeClassifier(),
```

```
    "Random Forest":RandomForestClassifier(),
    "Support Vector Machine": svm.SVC(),
    "K-Nearest Neighbors": KNeighborsClassifier(n_neighbors=3),
    "Multinomial Naive Bayes": MultinomialNB()
}

for i in range(len(list(models))):
    model = list(models.values())[i]
    model.fit(X_train,y_train.values.ravel()) # Train Model
    # Make predictions
    y_train_pred = model.predict(X_train)
    y_test_pred =  model.predict(X_test)

  # Test set performance
    model_test_accuracy = accuracy_score(y_test, y_test_pred)
    model_test_f1 = f1_score(y_test, y_test_pred, average='weighted')
    model_test_precision = precision_score(y_test, y_test_pred , average='weighted')
    model_test_recall  = recall_score(y_test, y_test_pred,average='weighted')

  # Training set performance
    model_train_accuracy = accuracy_score(y_train, y_train_pred)
    model_train_f1 = f1_score(y_train, y_train_pred, average= 'weighted')
    model_train_precision = precision_score(y_train, y_train_pred,average='weighted')
    model_train_recall = recall_score(y_train, y_train_pred,average='weighted')

    print(list(models.keys())[i])

    print('Model performance for Training set')
    print("- Accuracy: {:.4f}".format(model_train_accuracy))
    print('- F1 score: {:4f}'.format(model_train_f1))
    print('- Precision: {:4f}'.format(model_train_precision))
    print('- Recall: {:4f}'.format(model_train_recall))

    print('-----------------------------------')

    print('Model performance for Test set')
    print('- Accuracy: {:.4f}'.format(model_test_accuracy) )
    print('- Fl score: {:.4f}'.format(model_test_f1))
    print('- Precision: {:.4f}'.format(model_test_precision))
    print('- Recall: {:.4f}'.format(model_test_recall))


    print('='*35)
    print('\n')
```

```
Logisitic Regression
Model performance for Training set
- Accuracy: 0.8895
- F1 score: 0.888060
- Precision: 0.888324
- Recall: 0.889535
----------------------------------
Model performance for Test set
- Accuracy: 0.8837
- Fl score: 0.8821
- Precision: 0.8817
- Recall: 0.8837
===================================


Decision Tree
Model performance for Training set
- Accuracy: 1.0000
- F1 score: 1.000000
- Precision: 1.000000
- Recall: 1.000000
----------------------------------
Model performance for Test set
- Accuracy: 0.8372
- Fl score: 0.8391
- Precision: 0.8420
- Recall: 0.8372
===================================


Random Forest
Model performance for Training set
- Accuracy: 1.0000
- F1 score: 1.000000
- Precision: 1.000000
- Recall: 1.000000
----------------------------------
Model performance for Test set
- Accuracy: 0.8140
- Fl score: 0.8010
- Precision: 0.8066
- Recall: 0.8140
===================================
```

```
Support Vector Machine
Model performance for Training set
- Accuracy: 0.8605
- F1 score: 0.850455
- Precision: 0.872493
- Recall: 0.860465
----------------------------------
Model performance for Test set
- Accuracy: 0.7674
- Fl score: 0.7389
- Precision: 0.7511
- Recall: 0.7674
===================================


K-Nearest Neighbors
Model performance for Training set
- Accuracy: 0.9302
- F1 score: 0.928174
- Precision: 0.933976
- Recall: 0.930233
----------------------------------
Model performance for Test set
- Accuracy: 0.7907
- Fl score: 0.7710
- Precision: 0.7801
- Recall: 0.7907
===================================


Multinomial Naive Bayes
Model performance for Training set
- Accuracy: 0.8547
- F1 score: 0.847554
- Precision: 0.856964
- Recall: 0.854651
----------------------------------
Model performance for Test set
- Accuracy: 0.8605
- Fl score: 0.8507
- Precision: 0.8621
- Recall: 0.8605
===================================
```

```
C:\Users\student\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike ot
her reduction functions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis
it acts along. In SciPy 1.11.0, this behavior will change: the default value of `keepdims` will become False, t
he `axis` over which the statistic is taken will be eliminated, and the value None will no longer be accepted.
Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Users\student\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike ot
her reduction functions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis
it acts along. In SciPy 1.11.0, this behavior will change: the default value of `keepdims` will become False, t
he `axis` over which the statistic is taken will be eliminated, and the value None will no longer be accepted.
Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

In [ ]:

```
Conclusion.
When importing our base we can see that we have both categorical and continuous variables, we have a lot of col

Looking at the correlation we can see that there is no strong correlation between our data, when looking at our

When we compare our categorical variables with our Target variable, we can see that the Not Placed result is us

When we take the ssc_percentage variable to analyze, we can see that employees who are not from the central and

Talking about the machine learning models, we had to balance the classes, as in our database we have much more

Now talking about the most important variable for the machine learning models to reach the final result, it was
```